MASTER EN SCIENCES INFORMATIQUES

L'automatisation dans les centres de calcul

Verdure, Jean-Marc

Award date:
1988

Awarding institution:
Universite de Namur

Facultés Universitaires Notre-Dame de la Paix, Namur
Institut d'Informatique
Année Académique 1987-1988

L'AUTOMATISATION DANS LES CENTRES DE CALCUL

Jean-Marc VERDURE

Mémoire présenté en vue de l'obtention du grade de
Licencié et Maître en Informatique

R E S U M E

L'automatisation dans les centres de calcul

Ces dernières années, de profonds changements au niveau du matériel, du logiciel et des applications ont affecté les centres de calcul, les obligeant à repenser leur organisation face à de nouvelles exigences de qualité de service, de sécurité, de productivité...

Tous les domaines de l'exploitation sont touchés par une complexité croissante, les conditions de travail y sont déraisonnables, les erreurs fréquentes et coûteuses, les spécialistes rares et leur formation difficile. Face à ces problèmes, les gestionnaires des centres considèrent l'automatisation des tâches comme une nécessité.

Des outils logiciels existent depuis de nombreuses années, automatisant des aspects bien déterminés d'un système bien déterminé et survivant difficilement aux évolutions de sa configuration et de sa politique opérationnelle. De façon générale, ces outils classiques sont insuffisants et remportent un succès très mitigé auprès des exploitants.

Les systèmes experts (S.E.) semblent constituer une alternative intéressante dans la mesure où ils intègrent une connaissance et une flexibilité semblables à celles d'un expert. Des résultats encourageants ont déjà été obtenus dans le cadre d'applications comme YES/MVS, et l'on s'attend à ce que les S.E. jouent un rôle important dans l'exploitation des centres informatiques.

S U M M A R Y

Automation in computer centers

Last years, deep hardware and software changes affected the computer centers. As a result, they must reconsider their organisation to take up the new challenges of service quality, security and productivity.

All the operating areas suffered from an increasing complexity and poor working conditions. The errors are frequent, the experts rare and their training difficult. To face these problems, the managers are considering automation as a necessity.

For years, software companies are proposing tools automating specific areas of a specific system and enduring painfully configuration and operationnal policy changes. Generally, these tools aren't satisfactory and the computer center staff doesn't greet them with enthusiasm.

Expert systems (E.S.) seem to be an interesting alternative to the extent they integrate as knowledge and flexibility as a human expert. Conclusive results have already been obtained during the development of prototypes like YES/MVS, and E.S. are expected to play a leading role in the future.

A V A N T - P R O P O S

Le sujet de notre mémoire a été défini dans le cadre d'un stage pratique au centre de recherche en software de la société SIEMENS à Munich. Ce stage nous a permis d'appréhender les aspects "système" et "exploitation" et nous a sensibilisé aux problèmes qui forment la toile de fond du présent travail. Nous sommes dès lors très reconnaissants à la société SIEMENS SOFTWARE de Rhisnes de nous avoir permis d'acquérir cette expérience indispensable.

C'est aux membres de l'équipe SP 111 du centre de Munich que nous nous adresserons ensuite, tout spécialement Madame Kassel-Combaü et Messieurs Droletz, Linser et Spicker. Nous les remercions pour leurs multiples explications et l'important support documentaire qu'ils nous ont fournis, bien au-delà de notre stage.

Nous remercions enfin Monsieur Jean Ramaekers, promoteur de ce mémoire, et Madame Cécile Stas-Mahiat, son assistante, pour l'intérêt avec lequel ils ont suivi le développement de ce mémoire et pour les nombreux conseils qu'ils nous ont donnés.

T A B L E D E S M A T I E R E S

DEUXIEME PARTIE : L'AUTOMATISATION DANS LES CENTRES DE CALCUL

CHAPITRE 2 : VERS UNE AUTOMATISATION DES TACHES DANS LES CENTRES DE CALCUL

CHAPITRE 3 : INTERET ET ETAT DE L' AUTOMATISATION POUR DIVERS DOMAINES CLES

CHAPITRE 4 : L'APPROCHE "SYSTEME EXPERT"

INTRODUCTION

Dire actuellement de l'informatique, qu'elle va modifier considérablement la structure et la gestion des entreprises, relève du cliché... d'autant plus qu'il s'agit moins d'une prédiction de l'avenir que de l'observation d'une évolution. C'est dans le cadre de cette évolution que se situe notre mémoire, à l'heure où le système d'information devient pour l'entreprise un instrument stratégique.

Le but de notre travail est d'analyser les problèmes d'exploitation des centres informatiques induits par cette évolution, et d'évaluer l'apport de l'automatisation dans la résolution de ces problèmes. C'est donc la face cachée de l'informatique qui nous préoccupe, celle des centres de calcul, des opérateurs et des travaux batch, celle qui n'apparait à l'utilisateur final qu'en cas de troubles, et de plus en plus, celle qui doit améliorer la qualité de son service, accroître sa productivité et restreindre ses dépenses.

L'automatisation des tâches dans un centre de calcul est un sujet que l'on ne peut aborder de façon théorique. La plupart des métiers de l'exploitation ne s'apprennent d'ailleurs que "sur le tas". Avant d'entamer une étude sur le sujet, il est nécessaire de se familiariser avec les aspects 'système' et 'exploitation' d'un centre. L'expérience que nous avons gagnée dans le cadre de notre stage est bien modeste face à la complexité actuelle des centres informatiques, mais elle nous a sensibilisé aux problèmes des exploitants et nous a fourni certains prérequis nécessaires à l'étude de ce sujet. Nous relatons cette expérience dans la première partie de notre travail, mettant déjà en évidence un premier problème : l'exploitation d'un système informatique en l'absence d'un personnel spécialisé.

La seconde partie de notre mémoire traite de l'automatisation dans les centres de calcul. Quels sont les problèmes actuellement rencontrés dans les grands centres ? Quelles sont les nouvelles exigences qu'on leur impose ? Quels sont les défis auxquels ils devront faire face dans les dix années à venir ? En répondant à ces questions au chapitre 2, nous dressons le cadre général du problème.

Au chapitre 3, nous examinons l'intérêt que revêt une automatisation pour six domaines clés de l'exploitation. Pour chacun de ces domaines, nous décrivons les tâches impliquées, les difficultés éprouvées, l'intérêt d'une automatisation et les grandes tendances en matière d'outils logiciels. Dans un second temps, nous proposons un tableau comparatif des outils existants actuellement (du moins au niveau expérimental), précisant pour chacun d'eux le vendeur, les domaines concernés, le type, ainsi qu'une référence pour de plus amples informations. Devant l'interdépendance des tâches d'un centre de calcul, la multiplicité des outils et la similitude des données utilisées par ceux-ci, nous présentons finalement une architecture intégrée pour l'évaluation et le contrôle de la performance d'un système.

Parmi les outils que nous aurons rencontrés, les systèmes experts occupent une place honorable. Le chapitre 4 discute plus en détail l'approche "système expert" des problèmes d'exploitation. Après une introduction aux principes de base des systèmes experts et une justification de leur développement dans les domaines qui nous intéressent, nous présentons dans un second temps deux prototypes. Le premier, YES/MVS, assiste ou remplace un opérateur moyen dans ses fonctions non manuelles. Le second, TUNING AID, s'adresse plus spécifiquement au tuning d'un système d'exploitation UNIX.

Une représentation déclarative des connaissances est-elle toujours préférable à une représentation procédurale dans les domaines qui nous préoccupent ? Nous discuterons cette question dans un troisième temps.

Une approche globale de l'automatisation des centres de calcul ne semblant pas très réaliste actuellement, nous présentons dans un quatrième temps une technique qui pourrait être utilisée pour assurer l'intégration de divers systèmes experts spécialisés dans un domaine particulier de l'exploitation.

Nous terminons ce chapitre par une évaluation de l'approche "système expert", dégageant les forces et limitations de ces logiciels, ainsi que les perspectives qu'ils ouvrent dans le domaine de l'exploitation.

Notre mémoire veut avant tout appréhender le phénomène d'automatisation des centres de calcul de façon globale : sensibiliser le lecteur à une série de problèmes présents et à venir, dégager l'intérêt et la nécessité d'une automatisation, présenter les solutions classiques existantes et discuter des perspectives offertes par les systèmes experts.

Il n'existe pas à notre connaissance d'ouvrage traitant notre sujet dans son ensemble. Si les problèmes existent depuis longtemps dans les centres de calcul, le besoin d'une automatisation semble n'avoir été ressenti qu'assez récemment. Notre travail s'appuie sur de nombreux articles et comptes rendus de conventions informatiques, des discussions avec du personnel d'exploitation ainsi que des contacts avec d'importants consommateurs d'informatique.

PREMIERE PARTIE - UNE EXPERIENCE AU NIVEAU SYSTEME

CHAPITRE 1 : DESCRIPTION DE NOTRE ACTIVITE CHEZ SIEMENS

1.1 Introduction

Le département "D ST SP 1" du centre de recherche en Software de Siemens AG - Münich/Perlach (République Fédérale d'Allemagne) est responsable du développement du système d'exploitation BS 2000. Au sein de ce département, nous avons été adjoints au groupe SP 111, responsable de plusieurs outils de diagnostic, parmi lesquels, SLED (Self-loading Emergency Dump), SODA (Selective Organising-, Diagnosing- and Analysing Program), SNAP (Snapshot Dump) and CDUMP (User-, System- and Area-Dump).

Notre travail consistait à réaliser une nouvelle fonction dans SLED. Avant de décrire celle-ci, nous allons donner un aperçu de cet outil logiciel en le resituant dans son contexte : les dumps.

1.2 Les dumps

1.2.1 Généralités

La plupart des définitions relatives aux dumps (IFIP 1971), (ANSI 1970), (MAYNARD 1982), sont soit trop sommaires, soit trop spécifiques à un type de système. Celle que nous proposons a l'avantage d'être très générale.

Le dictionnaire Larousse Lexis (LAROUSSE 1977, p.560) définit un dump comme "une **image mémoire** dont **l'impression** est effectuée à l'occasion d'un **incident** pour rechercher une erreur". La définition de ce mot apparu vers 1970 se doit d'être nuancée et étendue. Précisons donc ce que nous entendons par image mémoire, incident et impression.

Une image mémoire ne désigne pas toujours une copie de la totalité de la mémoire. Comme nous allons le voir par la suite, l'étendue d'un dump est fortement variable. Dans le cas des systèmes ayant une mémoire à pagination virtuelle, le dump peut contenir, outre des portions de mémoire centrale, des pages de mémoire virtuelle se trouvant sur disque.

Un dump est toujours "tiré" dans un contexte d'incident. Les incidents considérés ici sont des suites d'erreurs hardware ou software et varient de la simple mise en suspens d'une tâche à la fin de session du système d'exploitation.

L'impression, quant à elle, ne peut être considérée comme un élément essentiel d'une définition. Un dump est en effet bien souvent "sorti" d'abord sur bande magnétique (ou disque) afin de réduire au maximum le temps de réquisition des ressources mémoires. Le contexte d'incident (précédemment cité) ne peut en effet souffrir le temps que nécessite la phase d'impression, d'autant plus que celle-ci doit être précédée d'une phase d'édition, d'organisation des données brutes.

Le but ultime d'un dump est la recherche (des raisons) d'une erreur : phases d'analyse et de diagnostic. Celles-ci sont confiées à des programmeurs système qui parcourent les données sous forme hexadécimale et examinent le contenu de différents registres et tables du système. Pour ce faire, ces experts disposent généralement d'un listing produit à partir des données brutes, à l'issue d'une phase d'édition. Cette phase d'édition et d'organisation des données est d'autant plus importante que le listing d'un dump peut atteindre plus de 60 centimètres de haut... le dump doit donc être structuré ! Il est également possible d'éviter le support papier en utilisant des outils interactifs qui travaillent sur les données brutes et fournissent directement l'information recherchée dans le dump. C'est le cas du produit DAMP développé chez SIEMENS.

En guise de récapitulation, nous pouvons distinguer 5 phases dans le cycle de vie d'un dump (cfr. figure 1.1).

1. **CREATION** : des parties de la mémoire sont copiées sur disque ou sur bande magnétique.
2. **EDITION** : le dump est organisé et édité en une forme lisible, représentation hexadécimale et caractères EBCDIC correspondants.
3. **IMPRESSION** : étape facultative quoique d'usage.
4. **ANALYSE** : analyse du dump, en parcourant le listing produit en phase 3 ou en consultant celui-ci au terminal au moyen d'un outil interactif. Ces outils n'exigent pas tous que le dump soit préalablement édité.
5. **DIAGNOSTIC** : détermination des causes de l'incident.

Les trois premières phases sont réalisées par différentes routines tandis que les deux dernières restent essentiellement manuelles... si toutefois nous pouvons qualifier ainsi des tâches qui requièrent pareille concentration et savoir-faire.

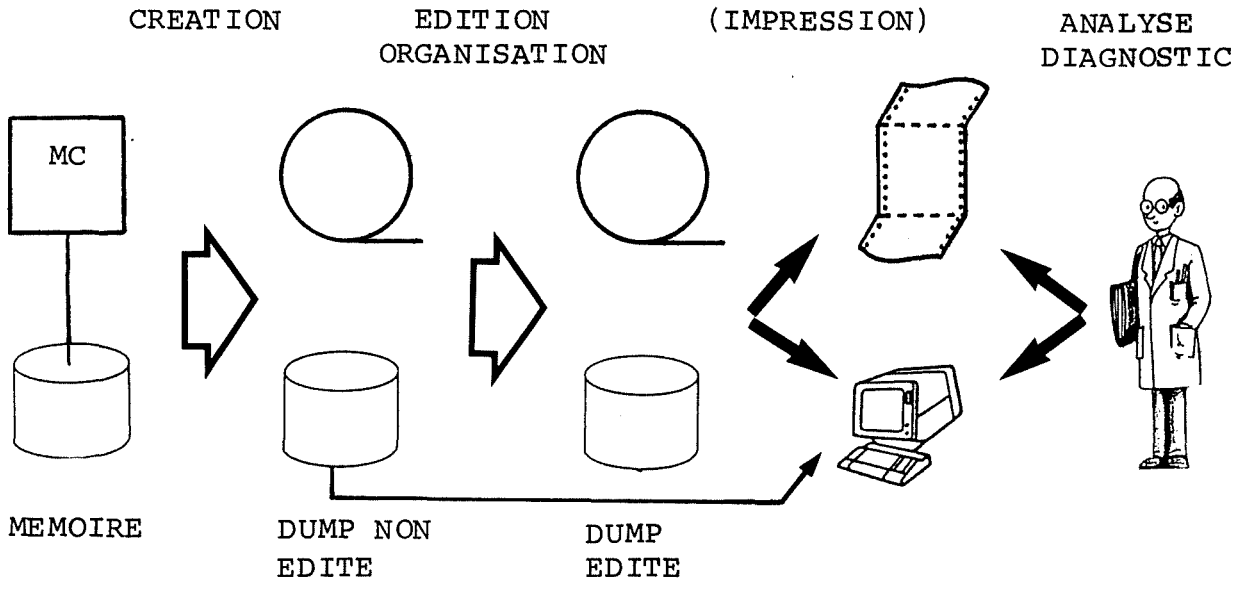

Figure 1.1 : Phases du cycle de vie d'un dump.

1.2.2 Classification des dumps chez SIEMENS

Chez Siemens, on peut considérer trois types de dumps produits par trois routines distinctes : CDUMP, SNAP et SLED.

CDUMP (Comfort dump)

CDUMP produit un dump relatif à une tâche quand une erreur irrécupérable survient dans un programme. La tâche concernée est arrêtée pendant le tirage du dump. La taille de celui-ci dépend des ressources du système utilisées et l'édition est réalisée grâce aux programmes SODUMP ou DAMP. On distingue deux catégories :

User Dump : l'appelant est non privilégié. La mémoire ainsi que les tables sorties ont uniquement trait aux ressources utilisées par l'appelant (ressources non privilégiées).

System Dump : l'appelant est une fonction du système. La mémoire et les tables sorties sont d'ordre privilégié, et correspondent à l'environnement de la tâche concernée (la fonction ayant demandé le dump travaille pour une tâche).

SNAP (Snapshot Dump)

SNAP produit une image instantanée du système. Dans le cas d'un tel dump, le système est arrêté pendant 12 secondes, la mémoire résidente du système est sortie sur disque et ensuite le système reprend son travail. Le dump sera par la suite copié et préparé avant d'être édité par SODA (routine d'édition) dans le cadre d'une tâche système. Cette phase de préparation qui s'ajoute ici relève d'un problème technique : le système ne peut être arrêté pendant plus de 12 secondes sans qu'une panne soit déclarée au niveau du réseau. Afin de ne pas dépasser ce délai, il est nécessaire d'écrire sur disque, cylindre par cylindre. Cette technique d'accès n'étant pas classique chez Siemens, elle n'est pas supportée par les routines d'édition, qui accèdent au disque piste par piste... d'où cette phase de préparation. La taille maximale est de 24 Mb.

SLED (Self Loading Emergency Dump routine)

Les deux types de dump considérés jusqu'ici ont un point en commun : ils sont "tirés" durant l'exploitation du système, qui est tout au plus interrompue pendant 12 secondes. SLED quant à lui est activé pour tirer un dump quand le système n'est plus en fonctionnement (postmortem dump). Avant de présenter cette routine plus en détail, nous donnons ici un tableau comparatif des différents types de dump chez SIEMENS (tableau 1.1). Les chiffres qui nous ont été communiqués par le centre de recherche de Munich ne sont pas valables pour tous les systèmes BS2000 et sont seulement donnés ici à titre de comparaison.

ROUTINE		TYPE DE DUMP	CONDITIONS D' EXECUTION	TEMPS D' EXECUTION	TAILLE DUMP	OUTPUT CLASSIQUE ET TAUX D'ECRITURE ASSOCIE		EDITION DU DUMP PRODUIT
CDUMP	System	system dump	arrêt momentanné de la tâche concernée	-	dépend des ressources utilisées	Disk	< 1 Mb/sec	DAMP SODUMP
	User	user dump						
SNAP		system dump	arrêt mom. système	<= 12 sec	<= 24 Mb	Disk	≈ 2,5 Mb/sec	SODA
SLED		system dump	arrêt déf. système	<= 30 min	<= 400 Mb	Tape	≈ 200-300 Kb/sec	SODA-AID-DAMP

Tableau 1.1 : Tableau comparatif des divers types de dump chez SIEMENS

1.3 Présentation et fonctions de SLED

SLED est exécuté comme un programme d'utilisateur privilégié sous le contrôle d' IPL-EXEC. Avant que SLED puisse être chargé, IPL-EXEC qui est une partie du programme de chargement initial (IPL : Initial Program Loader) doit lui-même être chargé et initialisé.

Quand il s'agit d'un premier chargement, les zones de mémoire utilisées par IPL et SLED doivent être sauvegardées. Ceci se fait en partie par le firmware en copiant les zones de données dans les "save areas" en mémoire ou vers le SVP (Service Processor) mais, pour la majeure partie, par software (en écrivant les zones de données sur le disque où réside IPL).

SLED offre deux fonctions, l'une dite d'édition (EDIT) qui n'est que très rarement utilisée et l'autre de non-édition (NOEDIT) .

Avec la fonction "NOEDIT", SLED écrit sur bande magnétique ou disque, un fichier de diagnostic (SLEDFILE), contenant toutes les données nécessaires à de futures analyses par une routine d'édition (SODA, AID, DAMP). Il s'agit principalement de portions de mémoire centrale et de mémoire virtuelle.

Avec la fonction "EDIT", SLED imprime les données éditées. Les données suivantes sont produites :

- status du programme avant le chargement de SLED
- contenu des différents registres spéciaux
- clés de protection de la mémoire
- tables de translation d'adresses (dans la mesure où elles sont cohérentes)
- contenu des zones de mémoire qui ont été sélectionnées par l'utilisateur

La figure 1.2 illustre une configuration souhaitable pour l'exécution de SLED (SIEMENS 1987 a).

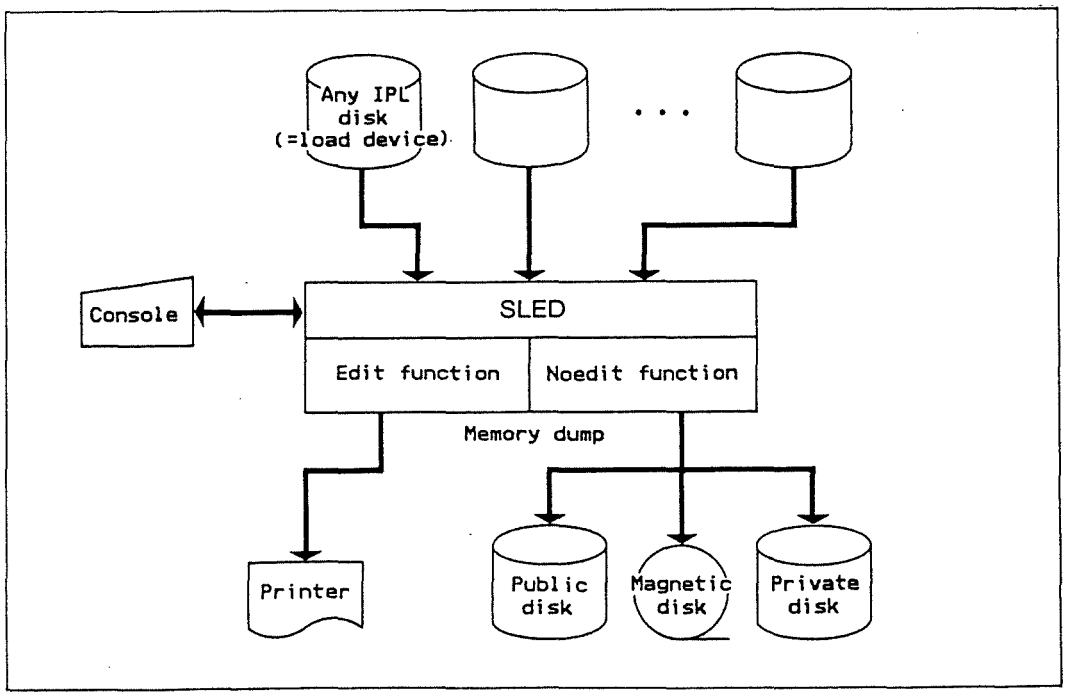

Figure 1.2 : Configuration souhaitable pour l'exécution de SLED (SIEMENS 1987 a)

1.4 Utilisation et objectifs de SLED

SLED est toujours activé lors d'une fin non planifiée d'une session du système. Les facteurs suivants peuvent être la cause d'une fin de session anormale :

- erreurs hardware non corrigibles au niveau du CPU ou des périphériques;
- erreurs software non corrigibles provoquées par des situations de deadlock ou de saturation;
- condition dans laquelle le système reconnait l'impossibilité de continuer un déroulement normal due à une incohérence totale des données;

Dans la plupart des cas, il est alors nécessaire de mémoriser l'état du système au moment de l'erreur avant de le relancer, afin de permettre un diagnostic ultérieur. SLED est activé à cette fin.

Notons également que SLED est utilisé par les programmeurs qui développent des applications "système". Un dump constitue en effet pour eux la seule aide au debugging disponible (cfr. 1.6).

1.5 SLED automatique et manuel

Nous avons déjà dit que SLED proposait deux fonctions qui se distinguent essentiellement par le périphérique de sortie utilisé. La spécification de ce premier paramètre sélectionne implicitement la fonction à exécuter. Ainsi, spécifier l'imprimante revient à choisir la fonction dite d'édition. D'autres paramètres doivent également être précisés tels :

- l'identifiant précis du périphérique;
- le nom du fichier qui contiendra le dump;
- l'étendue du dump;
- ...

Toutes ces valeurs qui permettent de contrôler SLED sont - en cas d'exécution manuelle - lues à la console, ce qui requiert la présence d'un opérateur. Celle-ci ne peut être postulée lors d'un "crash", événement par essence non planifié.

Une fonction "automatic restart" permet de démarrer automatiquement une nouvelle session du système. Dans ces cas d'automatic restarts, un dump doit également être "tiré". SLED peut à cette fin (et à la demande d'IPL) être exécuté en mode automatique. Actuellement, SLED utilise alors un ensemble de valeurs par défaut pour remplacer le dialogue avec l'opérateur. Ce système étant par trop rigide, il serait intéressant de pouvoir exercer un meilleur contrôle de SLED dans cet environnement de "restart" automatique. C'est à ce niveau que se situe notre collaboration. Nous allons maintenant la décrire tout en citant les avantages retirés de la solution adoptée, à court et moyen terme.

1.6 SLED et "Parameter File Facility"

La "parameter file facility" est une nouvelle fonction de SLED vers une meilleure flexibilité et une plus grande facilité d'utilisation. Cette nouvelle possibilité permet à un utilisateur de spécifier la fonction de SLED désirée, ainsi que les paramètres qui en dépendent, dans un fichier appelé "fichier de paramètres". Couplée avec une exécution manuelle de SLED, elle permet d'éviter à l'opérateur un long dialogue à la console et d'accélérer par la même occasion la reprise du système (le "restart" ne peut se faire qu'une fois le travail de SLED terminé). Couplée à une utilisation de SLED en mode automatique, notre nouvelle fonction offre la possibilité de contrôler l'exécution de SLED selon les circonstances et la configuration existante. Cette exécution automatique ne repose dès lors plus toujours sur le même ensemble de valeurs par défaut.

Ce nouveau fichier système est conçu et peut être modifié à tout instant par l'administrateur du système. Celui-ci peut ainsi mieux intégrer les besoins en ressources de SLED dans un planing global. La place nécessaire à un fichier de dump ne doit plus être réservée en permanence sur le même (groupe de) périphérique(s) et, en cas de goulet d'étranglement au niveau de l'espace total de stockage, l'étendue du dump peut être limitée. En cas de "restart" automatique, SLED, appelé par IPL en mode automatique, utilisera la dernière version en date du fichier de paramètres ayant le nom conventionnel "Sledpar".

Il est apparu très vite que ce mécanisme de lecture des paramètres dans un fichier pouvait être d'un grand intérêt pour les gens qui testent une nouvelle (version d'une) application. Ceux-ci terminent en effet bien souvent leur séance de test en "tirant" un dump qui leur permet d'établir un diagnostic des problèmes rencontrés. Un fichier de paramètres permet de les délivrer d'un dialogue intensif à la console; il leur suffit de préciser le nom de ce fichier.

On peut donc considérer quatre types de SLED suivant le mode, manuel ou automatique, et l'utilisation ou non d'un fichier de paramètres. La figure 1.3 résume les différents types d'exécution de SLED. En ce qui concerne le mode automatique, c'est une convention qui régit l'emploi d'un fichier de paramètres. Pour ce qui est du mode manuel, l'utilisateur peut spécifier son fichier de paramètres "filename", utiliser le fichier standard "STD", ou choisir un dialogue à la console.

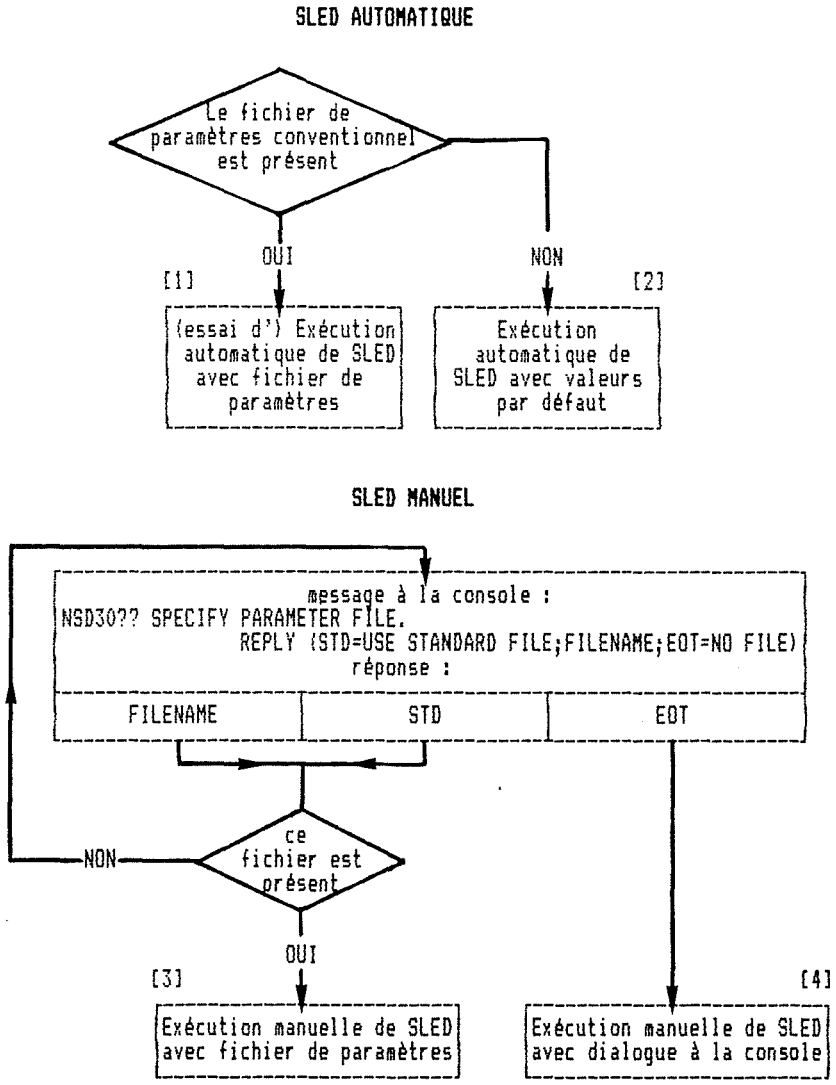

Figure 1.3 : Les 4 modes d'exécution de SLED

1.7 Implémentation de la "parameter file facility"

Les outils utilisés

L'implémentation de la "parameter file facility" a nécessité des modifications dans six des douze modules qui composent SLED, ce qui explique l'absence de listings en annexe... Nous avons réalisé ces modifications en F-Assembler, très semblable à l'Assembleur IBM 360/370 (mnémoniques identiques). Il est également d'usage de passer par le précompilateur COLUMBUS-ASS. Celui-ci simplifie la tâche des programmeurs en leur offrant des macros implémentant les structures de contrôle classiques : @if, @then, @else, @while, etc... et en les aidant

à régler les problèmes de passage de paramètres. Malgré tout, les conditions apparaissant dans ces structures de contrôle ainsi que la plupart des instructions doivent toujours être exprimées en Assembleur. COLUMBUS-ASS offre en outre des possibilités de documentation automatique : possibilités d'avoir le GNS du programme, un listing indenté avec mise en évidence des structures de contrôle par des traits verticaux, etc.

Les difficultés rencontrées

Si le principe de la "parameter file facility" peut s'expliquer en quelques mots, son implémentation n'est pas aussi aisée. Pour rappel, SLED est exécuté en l'absence du système d'exploitation et sous IPL-EXEC qui est une partie d'IPL, le programme de chargement initial. Nous ne pouvons donc pas utiliser les puissantes macros offertes par le "Data Management System" (DMS) de BS2000. Il ne s'agit pas non plus de manipuler des SVC (Supervisor calls), mais les macros que IPL-EXEC offre au programmeur sont tout de même d'assez bas niveau. Elles permettent par exemple d'accéder au bloc descripteur d'un fichier ou de lire un bloc de données de 2048 bytes. La vérification des caractéristiques d'un fichier de même que l'accès aux enregistrements qui composent celui-ci sont laissés à la discrétion du programmeur.

Implémenter la "parameter file facility" ne suppose aucun algorithme compliqué, aucune technique de programmation évoluée, seulement la maîtrise d'un environnement de très bas niveau et la résolution d'une succession de problèmes techniques.

Une difficulté tient à l'obligation d'assurer une compatibilité totale avec les versions précédentes. C'est une des raisons pour lesquelles nous n'avons pas pu réaliser une solution globale, qui consistait à isoler davantage l'acquisition des valeurs des paramètres des autres aspects. Pareille solution, quoique d'une réalisation plus difficile, nous semblait intéressante en vue de modifications futures. Evidemment "évolutivité" et "maintenance" n'ont pas été jusqu'ici les idées maîtresses du premier design de SLED... les modifications

ont touché six modules et dix procédures, dont neuf sans rapport direct avec le problème d'acquisition des paramètres.

Un autre problème tient à l'état de la documentation. Les commentaires face aux instructions Assembleur tiennent lieu de toute spécification et, si une description sommaire de la fonction réalisée par une procédure existe parfois en tête de celle-ci, elle n'a pas été mise à jour en fonction des modifications. Les programmeurs affirment d'ailleurs que seul le code dit ce que le module fait... Un travail important consiste donc à faire émerger les spécifications du code. Il existe évidemment des raisons d'ordre économique : sans rentrer dans le débat nous pouvons dire que SLED ne constitue pas la pierre d'angle de l'environnement BS2000 ! Il apparaît également, à l'issue de nombreuses discussions, que c'est là un problème général des applications du niveau système : de nombreux aspects reposent sur des conventions ou sur la mémoire des analystes et des programmeurs.

Gestion de versions

La gestion des différentes versions de SLED est réalisée par une technique proche de l'assemblage conditionnel. On transforme toujours les mêmes "modules pères" en utilisant une variable d'assemblage pour repérer les modifications relatives à un même objectif. Par exemple, nous avons choisi la variable "pff" pour repérer les modifications nécessaires à implémenter la "parameter file facility". Nous n'avons jamais effacé une partie du code d'un module père mais bien inhibé sa prise en compte conditionnellement à <pff=1>. De la même façon nous avons toujours inséré une nouvelle partie de code en rendant sa prise en compte conditionnelle à <pff=1>.

Lors d'une phase de pré-assemblage, on génère des modules fils relatifs à une même version en fixant la valeur de ces variables d'assemblage. Ces modules fils peuvent alors être traités par COLUMBUS-ASS puis assemblés. Par exemple, pour que les changements relatifs à la "parameter file facility" soient intégrés à la version 10, mais que les changements relatifs à la version 10.5 en préparation (et repérés par la

variable d'assemblage "105") soient omis, il faudra que les variables d'assemblage "pff" et "105" soient fixées respectivement à 1 et 0 avant de générer les modules fils.

Les tests

Faute de temps, nous n'avons pu tester l'efficacité de nos modifications. Nous avions prévu un plan de tests en deux phases : dans un premier temps, la nouvelle fonction devait être testée sur TESSIN, un programme capable de simuler le comportement d'un système SIEMENS ou IBM. Il n'était en effet pas possible de provoquer un crash du système pour notre bon plaisir. Dans un second temps seulement, cette fonction devait être testée sur un système réel.

1.8 Le recouvrement des erreurs

Si, avec un fichier de paramètres, SLED gagne une flexibilité certaine, le nombre de possibilités de "crash" au sein même de SLED s'en trouve également accru, ce qui suppose un système de recouvrement d'erreurs en conséquence.

Il y va de la responsabilité de l'administrateur du système ou de l'opérateur de remplir correctement le fichier de paramètres et de s'assurer des bonnes caractéristiques de celui-ci : fichier accessible en lecture, non protégé par un mot de passe, etc. De nombreuses erreurs sont cependant possibles et certaines d'entre elles ne peuvent être détectées qu'au moment de l'exécution. C'est le cas, notamment, lorsque suite à une reconfiguration, une unité disque ou un lecteur de bandes magnétiques n'existe plus.

Vu le contexte dans lequel SLED s'exécute, il s'agit de limiter au maximum le nombre de possibilités de "crash" internes. La nouvelle fonction introduite ne pouvant en aucun cas en induire davantage, des "switchs" d'un mode d'exécution à un autre sont prévus. La figure 1.4 donne une idée du traitement des erreurs lors de l'exécution de SLED. Pour chaque mode d'exécution (grand rectangle), il y a deux issues : NST (Normal SLED Termination) ou PROBLEM (erreur).

Si SLED détecte une erreur dans le fichier des paramètres - donnée manquante, incohérente ou invalide -, l'exécution est arrêtée et le fichier de paramètres abandonné. En cas de SLED automatique [1], l'exécution est alors reprise en utilisant le système de valeurs par défaut. En cas de SLED manuel [2], on demande à nouveau à l'utilisateur de choisir le type d'exécution manuelle qu'il désire. Il lui est alors loisible de spécifier un autre fichier de paramètres ou de choisir un dialogue à la console.

Un problème peut également survenir au cours d'une exécution de SLED en mode automatique avec valeurs par défaut [3]. Un passage à une exécution manuelle est alors prévu.

Deux types de problèmes peuvent survenir au cours d'une exécution manuelle avec lecture des paramètres à la console [4]. Certaines problèmes, comme une erreur dans la spécification d'un paramètre, peuvent être rattrapés, d'autres (plus rares) sont à tel point difficiles à régler que les concepteurs y ont renoncé. L'exécution de SLED est alors avortée, c'est ce que nous appelons ici un "crash".

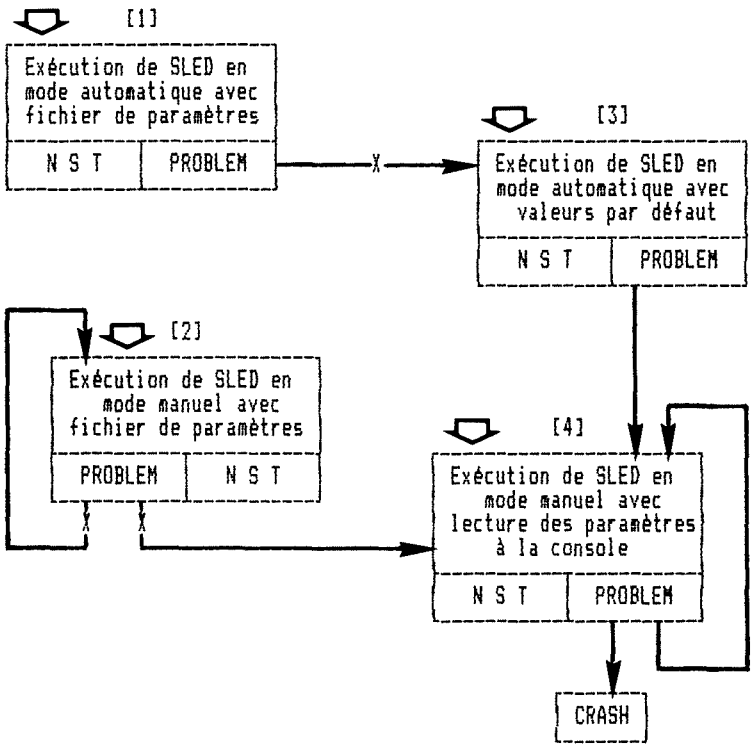

Figure 1.4 : Schéma général du recouvrement des erreurs lors de l'exécution de SLED.

Nous avons cité jusqu'ici les avantages directs retirés de la solution adoptée. On pense toutefois chez SIEMENS que la "Parameter File Facility" pourrait offrir d'autres avantages à moyen ou à long terme. Nous terminerons sur cette idée au point suivant.

1.9 Perspectives

Les "développeurs" de SLED seront d'ici quelques temps confrontés à un problème de taille... Afin d'aborder celui-ci, il est nécessaire de citer quelques chiffres (tableau 1.2). Les chiffres cités sont relatifs aux systèmes SIEMENS-BS2000 version 9.5. Cette version sera disponible pour les clients en 1989.

	MINIMUM	MAXIMUM
Mémoire centrale	4 Mb	512 Mb
Mémoire virtuelle	16 Mb	2048 Mb
Taille de la paging area sur disque		2000 Gb

Tableau 1.2 : Quelques chiffres relatifs aux systèmes BS2000 (version 9.5)

Le taux d'écriture d'un "Sled" (ainsi peut on appeler un dump produit par SLED) sur bande magnétique se situe entre 200 et 300 Kb/sec (principalement à cause des accès disque). Pour des raisons d'ordre pratique, un Sled doit être écrit en moins de 30 minutes, ce qui implique des tailles de dump de maximum 400 Mb. Pour les systèmes les plus importants (HAWK-B1 par exemple) un Sled recquiert actuellement 50 minutes environ. Certes, des systèmes de cette taille ne sont pas encore très courants si bien qu'un réglage de la performance de SLED est encore possible. Si les mémoires présentées au tableau 1.2 posent déjà problème pour SLED, l'avènement d'un nouveau "paging device" très performant appelé "**Expanded Storage**" va accroître considérablement les chiffres cités et... également les difficultés.

Une mesure simple qui peut être prise par les "développeurs" de SLED est de limiter la taille des dumps en ne "sortant" plus la mémoire utilisateur non privilégiée. Ceci n'est bien sûr pas suffisant à plus long terme. Les différentes tendances sont les suivantes :

- **meilleure qualité du logiciel système** afin de réduire le nombre de dumps;
- **dumps des données importantes** uniquement afin de réduire la taille des dumps;
- **meilleure utilisation des ressources hardware** (parallélisme) afin d'améliorer le taux d'écriture des dumps.

Relativisons quelque peu ces deux derniers points :

- En essayant de déterminer les données qui sont importantes et celles qui ne le sont pas, il est nécessaire de sacrifier la performance. En effet, discriminer les zones de mémoire importantes ralentit considérablement la création du dump , alors que le temps réservé à celle-ci est déjà limité. Le risque que toutes les données importantes ne soient pas "sorties" est donc plus important.
- En ce qui concerne le parallélisme, il est bon de noter qu'une routine de dump ne peut pas être trop complexe. Vu que c'est bien souvent le dernier recours pour obtenir du "matériel de diagnostic", cette routine **doit** fonctionner et, pour des raisons d'ordre économique, on ne peut envisager d'affecter autant de programmeurs au développement de SLED qu'on en affecte à celui de BS2000 !

Dans un avenir assez proche, on ne pourra plus envisager un dump complet. Une partie du travail réalisé 'at SODA time' (SODA est une routine d'édition qui réalise également une sélection et une première analyse des données) devra dès lors se faire 'at SLED time' (lors de l'étape de création), ce qui postule chez SLED davantage d'intelligence.

Une façon d'aborder ce problème serait d'ajouter dans SLED des paramètres qui spécifieraient les zones mémoire devant faire l'objet d'un dump. Ces zones, dépendant du contexte du "crash", seraient déterminées par ailleurs au niveau de divers composants du système. La "parameter file facility" constituerait alors un interface intéressant entre SLED et ces différents composants.

CONCLUSION DE LA PREMIERE PARTIE

Les motivations à la base du développement de la "parameter file facility" de SLED sont les suivantes :

- diminuer l'intervalle de temps entre deux sessions du système d'exploitation;
- éviter un dialogue intensif à la console pour le contrôle de SLED;
- permettre un contrôle flexible de SLED dans le cadre de l'environnement d'automatic restart de BS2000;
- à plus long terme, servir d'interface entre SLED et les composants du système qui causent les incidents, afin que ces derniers contrôlent eux-mêmes l'exécution de SLED.

La "parameter file facility" est un cas particulier d'un problème plus général : l'exploitation d'un système informatique en l'absence d'opérateur. Même si ce projet qui a piqué notre curiosité ne fait pas partie des plans à court terme de la société SIEMENS, notre stage nous a permis d'en appréhender l'intérêt et... la complexité.

Nous n'avions jusqu'ici qu'une vue utilisateur des systèmes informatiques. Notre stage nous a permis de passer à une vue "concepteur" et "exploitant" et nous a sensibilisé aux problèmes inhérents à la face cachée de l'informatique. Nous avons alors essayé de rechercher les causes de ces problèmes en les replaçant dans un cadre plus général (nous présentons le résultat de nos investigations au second chapitre).

Au cours de cette démarche, des tendances se sont dégagées qui semblent montrer que les difficultés des exploitations se multiplieront, et que l'automatisation des tâches constituera un élément essentiel de toute solution.

Vu notre intérêt croissant pour le sujet, nous nous sommes penchés sur les possibilités d'automatisation des tâches d'un centre de calcul. Si ces possibilités s'avèrent assez nombreuses, la plupart des centres restent malgré tout très attachés aux procédures manuelles... Notre étonnement face à l'état du développement de l'automatisation dans les centres de calcul est à la base de notre mémoire.

DEUXIEME PARTIE - L'AUTOMATISATION DANS LES CENTRES DE CALCUL

CHAPITRE 2 : VERS UNE AUTOMATISATION DES TACHES DANS LES CENTRES DE CALCUL

2.1 Introduction

Les cinq dernières années ont vu des modifications profondes intervenir au sein des exploitations et l'on peut s'attendre à une répétition de celles-ci dans les dix années à venir...

Ces changements se situent à la fois au niveau du matériel utilisé, du logiciel et des applications. Ils sont en train de modifier considérablement les rapports entre utilisateurs d'informatique et informaticiens, obligeant ces derniers à améliorer la qualité des services offerts, à remettre en question le rôle des centres informatiques (C.I.) et à revoir radicalement leur organisation. Ceux-ci sont en effet appelés à devenir de vraies "centrales de fourniture d'énergie informatique" dans les entreprises.

Les exploitants des C.I. sont de plus en plus confrontés à une croissance rapide des besoins, à une multiplicité de nouveaux matériels et logiciels ainsi qu'à des impératifs de rentabilité. Face à cette complexité croissante de la mission des C.I. , ils considèrent l'automatisation comme un élément essentiel de toute stratégie. Selon A. RIVIERE, ingénieur informaticien chez IBM France, il s'agit moins de maîtriser la situation existante que de préparer l'évolution future (RIVIERE 1983).

Après une brève description de la tâche du personnel d'exploitation, nous rendrons compte de l'évolution qui se dessine depuis quelques temps déjà dans les centres informatiques... l'automatisation apparaissant alors comme une conclusion logique de cette évolution.

2.2 Le personnel d'un centre de calcul

2.2.1 Fonctions d'hier, d'aujourd'hui et de demain

A l'époque où le batch était le seul mode d'exploitation possible, l'accès à l'ordinateur était réservé à un personnel spécialisé : les pupitreurs et les opérateurs (RAMAEKERS 1985).

- **Les pupitreurs** seuls ont accès à l'unique terminal interactif de la configuration, le pupitre, où apparaissent tous les messages d'exception envoyés par les routines d'interruption lors d'incidents et à partir duquel ils dialoguent pour assurer l'avancement des travaux.
- **Les opérateurs** assurent les tâches subalternes de placement des paquets de cartes au lecteur, de montage et démontage des bandes magnétiques aux dérouleurs, de retrait des paquets de cartes au perforateur et des états imprimés à l'imprimante rapide. Certains auteurs emploient le terme de **préparateur** pour désigner la même fonction.

Ce mode d'exploitation suppose également la présence de **perforatrices** de cartes et d'une autre personne, que nous appellerons l'**administrateur du système** dirigeant tout ce personnel vers la réalisation d'un objectif commun: la gestion des ressources informatiques.

L'exploitation d'un ordinateur est aujourd'hui plus souple et repose moins sur la présence d'un personnel spécialisé. La fonction de perforateur a quasiment disparu, de même que la distinction entre pupitreur et opérateur. Le terme de pupitreur n'est d'ailleurs pratiquement plus employé. Quant à la frontière entre les deux fonctions restantes : opérateur et administrateur système... elle reste bien élastique ! Nous décrirons ces fonctions aux points 2.2.2 et 2.2.3. Bornons-nous

simplement à dire maintenant que théoriquement :

- **l'administrateur du système** réalise une gestion à moyen et long terme et garantit un certain niveau de service aux utilisateurs;
- **l'opérateur** s'occupe de l'exploitation journalière du centre.

La charge associée à chacune de ces fonctions varie fortement d'une organisation à l'autre. Les opérateurs doivent parfois réaliser des fonctions de "system management" et les administrateurs du système sont parfois concernés par des fonctions d' opérateur. On peut dire que "l'administrateur du système définit un cadre pour l'allocation des ressources et l'utilisation du système et que l'opérateur prend des mesures dans ce cadre" (DIGITAL 1986 b). La distinction est si peu marquée entre ces deux fonctions que bien des auteurs y renoncent, regroupant les opérateurs et les administrateurs système sous le vocable "operations staff" (DIGITAL 1986 b). Dans la suite de ce travail, nous parlerons de **politique opérationnelle** pour signifier les décisions de l'administrateur du système, et nous utiliserons le terme d'**opérateur** pour désigner le responsable de l'exploitation journalière du centre. Nous parlerons également des **opérations** pour désigner le travail de l'opérateur.

Notons tout de même, pour la complétude, que dans certains centres de calcul très importants, la répartition des fonctions est souvent plus fine et l'on distingue les profils suivants (Monnin 1985).

- **Ingénieur système**: est responsable de la mise en place, de l'utilisation effective, et de la maintenance des systèmes d'exploitation ainsi que des logiciels de base. Il a, en principe, sous ses ordres des analystes et programmeurs système et peut avoir des responsabilités supplémentaires en matière de support technique.
- **Analyste système** : effectue des travaux dans les domaines des systèmes d'exploitation et des logiciels de base. Il peut être chargé de la réalisation de logiciels "sur mesure" et être en contact avec les constructeurs et les SSII. Dans les petits et moyens centres, il arrive qu'il soit le seul spécialiste système.

- **Programmeur système** : réalise également des travaux dans les domaines des systèmes d'exploitation et des logiciels de base, et en rend compte à l'ingénieur système. Mais il n'a pas la même expérience que l'analyste système.
- **Responsable des méthodes d'exploitation** : est responsable de la définition des normes et supports d'utilisation des ressources. Il a sous ses ordres des spécialistes des méthodes d'exploitation et a généralement acquis une expérience à ce niveau.
- **Spécialiste des méthodes d'exploitation** : a la charge de résoudre les problèmes d'exploitation et d'assurer les reprises (c'est un "dépanneur en tout genre"). Il a normalement acquis une grande expérience de l'exploitation.

La fonction de perforatrice a disparu suite à une évolution technique et l'on peut également s'attendre à ce que la fonction d'opérateur subisse le même sort, suite à une automatisation poussée des centres de calcul. De nouvelles fonctions sont également appelées à se développer. Celles-ci dérivent des anciennes et se caractérisent par un rapprochement de l'informaticien et de l'utilisateur au moment où la micro-informatique s'installe dans les bureaux. Il s'agit des fonctions d'assistant, de promoteur et de conseil.

- Les **assistants** aident les utilisateurs à bien se servir des nouveaux produits. Ce sont souvent d'anciens pupitreurs ou opérateurs. Wachspress emploie le terme d'**installateurs de produits (WACHSPRESS 1986).**
- Les **promoteurs** aident les utilisateurs qui "programment" eux-mêmes leurs applications à bien exprimer et formaliser leurs besoins. Ce sont des analystes-programmeurs.
- Les **conseils** s'occupent des problèmes de connexions et autres problèmes techniques imposés par la multiplication des micro-ordinateurs et des appareils de saisie portables. Ce sont des techniciens ou des gens ayant exercé des fonctions dites "système".

La constatation de cette évolution au niveau du personnel doit poser aux responsables des centres informatiques la question de la reconversion progressive du personnel d'exploi-

tation. "Tel centre qui, il y a cinq ans fonctionnait 5,5 jours par semaine avec 40 personnes emploie aujourd'hui 16 personnes pour une activité ininterrompue 7 jours sur 7; dans cinq ans, le même centre n'emploiera plus que 8 personnes, installateurs de produits et contrôleurs polyvalents (WACHSPRESS 1986)".

Nous allons maintenant donner une idée du travail d'un administrateur de système et décrire ensuite plus en détail celui de l'opérateur puisque c'est actuellement l'homme clé du centre de calcul et que nous réserverons une large part de notre mémoire à discuter de l'automatisation de ses fonctions.

2.2.2 Le travail d'un administrateur de système

L'administrateur d'un système réalise la gestion du système à moyen et à long terme. A moyen terme, c'est lui qui autorise l'accès des utilisateurs (création de "directories"), leur accorde les ressources nécessaires (détermination des "quotas" disque) et gère leurs comptes (facturation des temps machines). Il doit également s'assurer que les différentes ressources sont utilisées à bon escient et éviter toute forme de gaspillage. C'est également à lui de veiller à l'efficacité des opérations du système (utilisation optimale et maintenance des ressources hardware, backups réguliers,...). A plus long terme, l'administrateur doit maintenir le système en fonctionnement (start, shutdown, restart), surveiller sa performance, sa sécurité et sa capacité, s'assurer de la satisfaction des utilisateurs, prendre les mesures destinées à améliorer la performance du système et la qualité du service, planifier l'achat de nouveau matériel, etc.

2.2.3 Le travail d'un opérateur

Le travail d'un opérateur est assez varié. Il réalise un certain nombre de tâches de routine comme monter une bande magnétique, changer le format du papier d'une imprimante, répondre au téléphone et surveiller activement l'état du système en observant, sur une ou plusieurs consoles, les messages émis par celui-ci. Il a des contacts privilégiés avec les utilisa-

teurs, car il constitue toujours pour eux une première aide, ainsi qu'avec les techniciens, qu'il guide afin de favoriser l'interprétation des problèmes et d'accélérer leur résolution.

La plupart des actions d'un opérateur peuvent être divisées en trois catégories : monitoring, allocation des ressources et traitement des problèmes. La latitude dont dispose un opérateur dans chacun de ces domaines varie fortement d'un type de système à un autre. L'allocation des ressources, par exemple, est un processus très largement automatisé dans certains systèmes. Dans une volonté d'être assez général, nous avons préalablement pris connaissance du travail d'un opérateur pour des systèmes aussi différents que VAX/VMS de DIGITAL (DIGITAL 1986 b, DIGITAL 1986 c) et OS/VS2 MVS d'IBM (MILLIKEN 1986). Nous ne tiendrons pas compte ici des différents "outils" pouvant simplifier la tâche de l'opérateur. Ceux-ci seront envisagés au chapitre 3.

Monitoring

Si les ressources sont trop sollicitées par les utilisateurs, la performance du système peut devenir mauvaise. L'opérateur doit donc surveiller l'activité du système en rassemblant des informations concernant les processus en cours, la mémoire, les files d'attente, les périphériques, les utilisateurs et l'utilisation des ressources. Il identifie ainsi des tendances et détecte des problèmes soit en comparant les valeurs obtenues à des seuils critiques, soit en utilisant des moyens plus complexes quand l'information historique doit être considérée. Une fois les problèmes mis en évidence, l'opérateur prend des actions en vue d'ajuster le système. Si le système se ralentit, à cause d'un manque de ressources par exemple, il peut limiter le nombre de processus batch ou interactifs, il peut réduire le nombre d'utilisateurs simultanés voire même exiger de certains un "log off". Une des difficultés qu'un opérateur rencontre dans sa tâche de monitoring est la nécessité de maintenir mentalement un modèle conceptuel du système. Il doit en effet savoir à quel rythme le statut des ressources peut changer et à quel point celles-ci sont critiques pour les services fournis, afin de déterminer les questions à poser au

système ainsi que la fréquence de ces interrogations.

Allocation des ressources

Les deux tâches principales sont ici la planification des travaux et la fourniture manuelle des ressources nécessaires aux entrées/sorties : (dé)montage des bandes magnétiques, chargement du papier aux imprimantes, etc. Suite à l'évolution de la charge, l'opérateur peut également être amené à modifier certains paramètres du système (tuning). La responsabilité d'un opérateur en ce qui concerne la planification des travaux varie fortement avec la politique du centre. En principe, c'est l'administrateur du système qui choisit les critères d'ordonnancement des files d'attente et le travail de l'opérateur consiste alors à maintenir ces files selon les priorités fixées. Bien souvent, suite à un monitoring, ce dernier est amené à modifier l'ordonnancement par défaut pour l'ajuster à la charge, aux ressources disponibles et aux circonstances inhabituelles. Ici aussi, l'opérateur doit avoir une bonne connaissance de la charge, des priorités, des dates limites, des ressources disponibles et demandées, etc.

Traitement des problèmes

Les problèmes sont généralement approchés de la manière suivante.

1. Identifier le problème aussi précisément que possible.
2. Rassembler les informations importantes dans le cadre du problème, en questionnant le système.
3. Etablir un diagnostic en s'aidant des manuels existants, de son expérience et des autres facilités offertes comme le télédépannage.
4. Prendre des actions pour résoudre le problème. Il s'agit souvent d'une séquence de commandes et éventuellement d'interventions physiques comme l'allumage d'un périphérique, la suppression d'un bourrage à l'imprimante, etc. Quand l'opérateur doit faire face à des problèmes complexes et inhabituels, il procède souvent par coups d'essais et erreurs cherchant des solutions dans un "bag of tricks".

5. Afin de s'assurer que le problème a été correctement résolu et qu'il ne se reproduira pas trop vite, accroître le monitoring.
6. Reporter l'incident dans le journal de bord.

Le problème principal est ici la pression exercée sur les opérateurs pour diagnostiquer et résoudre le problème rapidement. Cette pression combinée à la complexité inhérente au processus pousse parfois les opérateurs à commettre des erreurs qui peuvent aggraver la situation.

Gabriel Lestrate, du centre de calcul des Facultés Universitaires Notre-Dame de la Paix à Namur note, à cet égard, l'importance de la prévention en matière de problèmes "hardwares". Il vérifie régulièrement le bon fonctionnement du matériel et observe à la console les messages d'erreurs systèmes concernant les périphériques (erreurs au niveau des disques et dérouleurs de bandes). En observant l'évolution de ces erreurs, il peut prévenir une panne voire même une catastrophe.

La complexité inhérente au travail d'un opérateur

Nous avons déjà mis en évidence une première difficulté du travail d'un opérateur, la nécessité de maintenir mentalement un modèle du système. Les opérateurs ne basent en effet pas leurs actions sur un seul message. Au contraire, ils retiennent de nombreuses informations.
- Statut du système (valeurs courantes de ses paramètres, charge, statut des périphériques d'entrée/sortie).
- Fiabilité du modèle qu'ils ont du système. Les opérateurs doivent savoir quand les informations qu'ils possèdent doivent être rafraichies.
- Statut de la tâche ou du problème en cours.

Un second problème provient du fait que l'opérateur est toujours appelé en dernier ressort pour garder le système en état de marche. Quand un programmeur qui développe un logiciel rencontre une situation extrêmement complexe (ou nécessitant des connaissances relevant de la politique opération-

nelle), il n'a pas d'autre choix que d'y insérer un message à l'opérateur. L'opérateur doit ainsi résoudre des problèmes que les développeurs ont été incapables d'attaquer. Reporter un problème au niveau de l'opérateur se justifie dans certains cas mais il existe aussi de nombreux abus.

2.3 Evolution dans les centres informatiques

2.3.1 Le matériel, les logiciels, les applications

Ces dernières années, la mise en oeuvre des ressources informatiques au sein des entreprises s'est considérablement transformée. A côté du gros système centralisé gérant un réseau de terminaux esclaves, apparaissent des réseaux de mini-ordinateurs tandis que les micro-ordinateurs s'imposent chez un grand nombre de gestionnaires.

La communauté européenne estime qu'il y avait en 1984 sur le vieux continent six millions de micro-ordinateurs professionnels, et qu'il devrait y en avoir quarante millions en 1990. Celà signifie qu'en l'an 2000, l'informatique individuelle représentera 75% de la puissance informatique totale installée alors qu'elle n'en représentait que 35% en 1982 (DUBOIS Y. 1988).

Cet engouement pour les micros au détriment des "mainframes" peut également s'observer chez les étudiants, à l'institut d'informatique. J. TAMINE, chef du service Informatique et Organisation à la Société Carbochimique (Tertre - Belgique) note que l'on parle de plus en plus d'une informatique légère et peu coûteuse, "...l'euphorie qui régnait au début de l'ère des minis transparait à nouveau : petites machines, petits prix, petits ennuis ! (TAMINE 1983)". W. Wachspress, président général de la SISRO , dans un article intitulé "l'ère des automates" va jusqu'à poser la question de l'existence même des grands centres de calcul: "ne sont-ils pas voués à l'extinction, comme les dinosaures de l'ère secondaire ?". Selon lui, l'importance relative des grands centres tels que nous les connaissons aujourd'hui va décroître à cause de l'explosion permanente des techniques. Mais, en valeur absolue,

l'activité de ces centres va continuer de se développer à raison de 30 à 40 % l'an (WACHSPRESS 1986).

Les services d'exploitation ne sont pas les seuls touchés. En effet, des logiciels (ou progiciels) aux fonctions très générales et d'usage très simple sont actuellement disponibles sur la plupart des ordinateurs personnels : traitements de texte, tableurs, gestionnaires d'écran, systèmes de gestion de bases de données, langages d'interrogation, systèmes d'aide à la décision,... On s'attend à l'avenir à un accroissement sensible de l'usage de progiciels applicatifs au détriment des produits "maison" (WACHSPRESS 1986). Il faudra également compter avec les utilisateurs qui, à l'aide de langages de haut niveau, développeront leurs propres programmes.

De notre point de vue cette évolution est importante, et ce, pour trois raisons.
Premièrement, son incidence sur l'organisation du centre informatique est considérable. Comme nous allons le voir, la responsabilité du centre ne se cantonne plus à un site central mais s'étend à un nombre souvent important d'installations décentralisées.
Deuxièmement, des utilisateurs influencés par des applications micro ou vidéotex vont exiger du système central une qualité de service, une disponibilité, une sécurité sans cesse meilleure ainsi que des structures de coût plus légères.
Troisièmement, le personnel d'un centre devra assurer des fonctions d'assistance, de promotion et de conseil (cfr. 2.2.1).

Cette évolution marquante, qu'un grand nombre n'hésite pas à qualifier de révolution, ne doit pas en faire oublier une autre, tout aussi significative quoique moins spectaculaire : il s'agit de l'évolution de la demande en informatique. Les directions d'entreprises considérant de plus en plus l'information comme une ressource à part entière, au même titre que le capital et le travail, l'outil "système d'information" devient un élément essentiel de toute stratégie. Lors de la Convention Informatique à Paris, en 1986, voici ce que l'on prévoyait pour les grands centres de calcul de la prochaine décénnie (WACHSPRESS 1986) :
"Des processeurs de plusieurs dizaines de MIPS, des mémoires de

masse décuplées, plusieurs milliers de terminaux actifs ou passifs, des centaines de micro-connectés par vacation ou en permanence... ...une activité permanente, 24 heures sur 24, tous les jours. Applications diversifiées , du batch traditionnel, toujours vivace, au transactionnel ; plus de communication, plus de travaux à la demande...".
Face à ces affirmations que personne ne considère plus comme des éléments de fiction, on conçoit aisément que la tâche du personnel d'un centre n'ira pas en se simplifiant...

Toutes ces évolutions, au niveau du matériel et du logiciel ne manquent pas de provoquer des bouleversements dans l'organisation des centres informatiques. Avant de décrire ceux-ci, penchons-nous sur une autre cause de ces bouleversements : une évolution certaine des rapports de force entre les utilisateurs d'informatique et les informaticiens...

2.3.2 Les rapports avec les utilisateurs

Une annonce assez récente pour un ouvrage sur la gestion des stocks carricature assez bien la situation passée en parlant du "directeur informatique tout puissant, qui régnait sur d'énormes machines et annonçait un délai de trois ans lorsqu'un cadre, forcément subalterne, osait demander l'étude d'un programme qu'il avait eu l'inconvenance de juger important". L'informatique a en effet été bien longtemps une machinerie lourde et difficile à mettre en oeuvre, autour de laquelle gravitait une véritable caste d'informaticiens.

La situation actuelle est bien différente et repose sur la constatation que l'informatique n'est pas un univers à part, non soumis aux règles de l'ensemble de l'entreprise. Le département Informatique est constamment tenu d'améliorer sa productivité ainsi que la qualité de son service. Par qualité de service, on entend plus seulement une obligation de moyens (création d'un infocentre par exemple) mais également une obligation de résultats, ce qui oblige les directions informatiques à suivre leur gestion sous deux angles : suivi de leur budget et suivi de la satisfaction des utilisateurs.

Nous assistons donc à une inversion des rôles. "Alors que l'informaticien se voyait reprocher d'ignorer superbement l'utilisateur, de plus en plus l'utilisateur ignore l'informaticien (MOREAU 1988)". L'utilisateur ne raisonne plus en terme de machines mais de besoins à satisfaire, de traitements à effectuer, de temps de réponse à garantir et de liaisons à obtenir avec tout membre de l'organisation avec qui il peut avoir besoin de communiquer. "Peu importe qui lui fournit le moyen de satisfaire ses besoins. Le matériel et le système informatique ne sont plus que ces moyens, d'autant plus répartis que sont maîtrisées les techniques des systèmes distribués (MOREAU 1988)".

2.3.3 Les problèmes d'organisation

Comme nous avons eu l'occasion de le souligner jusqu'ici, les C.I. ont à remplir une mission de plus en plus complexe et doivent repenser leur organisation face à un certain nombre de nouvelles exigences : décentralisation des moyens informatiques, intensification de l'exploitation, sécurité et souplesse d'utilisation, qualité du service et rigueur, productivité,...

Décentralisation des moyens informatiques

Tant que les moyens informatiques se bornaient à un ordinateur central nécessitant la présence d'un personnel spécialisé, le personnel d'exploitation conservait toutes ses prérogatives et aucun problème d'organisation majeur n'était à déplorer. Avec l'avènement des minis et micros, machines capables de fonctionner "seules", la situation est devenue différente. Diverses philosophies d'organisation sont envisageables :

- le département informatique continue à gérer le système centralisé et le logiciel correspondant, et ignore le développement d'une "informatique parallèle";
- le département informatique veut rester maître du jeu et canalise le phénomène micro en créant un infocentre (pour plus de détails : BRANCHEAU 1985);

- le département informatique assume l'évolution et fournit des services supplémentaires d'assistance et de conseil aux utilisateurs de micro-informatique;
- ...

Il existe bien sûr un continuum de solutions entre ces diverses philosophies et l'adoption d'un type d'organisation dépendra de nombreux facteurs. Parmi eux :

- La position de force de l'informatique au sein de l'organisation. Difficile en effet d'envisager une solution globale dans les entreprises où l'on peut justifier l'achat d'un micro-ordinateur par note de frais !
- Les rapports de force entre informaticiens et utilisateurs ainsi que la mentalité de ces derniers. En effet, .l'idée que l'informatique n'est plus le fait de quelques initiés, le désir d'une informatique moins dépendante, plus aisément maîtrisable, sont autant d'arguments en faveur d'une décentralisation de l'organisation !
- La disponibilité du personnel informatique pour assurer les nouvelles fonctions d'assistance, de promotion et de conseil.

Notons que la solution qui consiste à ignorer le développement de la micro-informatique ne semble pas très réaliste à longue échéance. Il ne s'agit en effet pas seulement d'une évolution technique. L'ordinateur dit personnel porte en lui un modèle de décentralisation qui correspond bien à l'air du temps. Pour René Moreau, l'informatique personnelle va indubitablement diminuer le rôle des centres de calcul dont les tâches se trouveront morcelées à travers les différents postes de travail de l'entreprise. A l'avenir, ces centres ne se verront plus confier que les travaux répétitifs comme la paye, la facturation, etc. Quant aux bases de données centrales de plus en plus vastes, on peut s'attendre à ce qu'elles soient davantage réparties... si l'évolution de la technologie le permet (MOREAU 1988).

Intensification de l'exploitation

L'intensification de l'exploitation résulte de la concurrence de deux facteurs déjà évoqués plus haut : l'accroissement de la demande en terme de traitement d'informations et l'exigence de rentabilité des C.I. qui se doivent de contrôler et de limiter leurs coûts de fonctionnement. Nous avons déjà mis en évidence la complexité inhérente aux fonctions d'opérateur et d'administrateur de système, ce phénomène d'intensification touche à la fois les administrateurs des C.I. et le personnel chargé de l'exploitation journalière du centre, complexifiant encore leur tâche.

Du côté de l'exploitation nous pouvons remarquer que, malgré une tendance à une utilisation interactive des ordinateurs, le volume des travaux batch reste important... on peut d'ailleurs s'attendre à une augmentation de 30 à 40 % l'an au cours des dix prochaines années (BEAVER 1985). Le scheduling de ces travaux pose de plus en plus de problèmes.

- Ils doivent être accomplis dans des délais souvent assez brefs.
- Exécuter ces travaux "hors journée" n'est ni toujours possible (ceci nécessite en effet un travail en double ou triple équipe), ni toujours suffisant. "La journée de 24 heures est une limite perçue de plus en plus comme une contrainte, l'intégration des applications étant de plus en plus grande et les liens inter-applications de plus en plus présents" (MINGER 1986).
- Un réordonnancement continu des demandes de travaux est nécessaire afin de remplir les objectifs de priorité fixés.
- Un temps de réponse valable doit malgré tout être garanti aux utilisateurs interactifs.

Une autre conséquence remarquable de l'intensification de l'exploitation est l'accroissement considérable du nombre de messages apparaissant à la console. Il s'agit d'un problème aigu que nous avons eu l'occasion de percevoir dans le cadre de notre stage ainsi qu'au cours de discussions avec de gros consommateurs d'informatique (EXXON Chemicals, Royale Belge,...). Il s'agit bien d'un problème... dialoguer une heure

à la console dans un grand centre de calcul est en effet une expérience éprouvante, le nombre de messages par seconde étant tout à fait inconfortable. On remarque également que le pourcentage de messages importants du point de vue d'une exploitation immédiate est peu élevé.

Même si, dans un premier temps, il est possible de répondre à la demande croissante de traitement d'informations en intensifiant l'exploitation, les grands centres de calcul doivent bien souvent finir par installer des clusters de processeurs. Cette distribution des ressources rend les systèmes plus complexes à bien des points de vue, notamment du point de vue des tâches opérationnelles.

Du côté de l'administration du C.I., l'intensification rend plus complexes les tâches de planification des besoins et de suivi des projets. Les besoins en personnel, en puissance de calcul et en logiciels doivent être soigneusement planifiés en vue de pouvoir justifier une augmentation de capacité, le recrutement de personnel supplémentaire ou le développement d'un nouveau projet. L'informatique coûte en effet assez cher et un gestionnaire veut prendre des décisions de manière objective. L'administrateur du C.I. doit donc appuyer ses requêtes en se basant sur l'évolution de la charge et du temps de réponse, le nombre et la satisfaction des utilisateurs, l'avancement des projets, d'éventuelles économies attendues,...

Sécurité et souplesse d'utilisation

La sécurité est une exigence qui se pose de plus en plus à toute organisation. Si cette attention a été quelque peu négligée dans les premières années de l'apparition des ordinateurs, il n'est plus envisageable aujourd'hui de ne pas inclure des contraintes de sécurité dans les spécifications d'exploitation ou de programmation. La sécurité en informatique présente en effet une nature particulière car l'informatique se caractérise par la concentration des informations et des procédures codées de traitement de celles-ci. Un centre informatique doit envisager la sécurité sous trois aspects.

1. Garantie d'intégrité : Le demandeur de traitement doit avoir la garantie que les données exploitées sont bien celles qu'il croit, qu'elles se trouvent dans un état normal et qu'elles sont exploitées selon les procédures qu'il a définies.
2. Garantie de secret : assurer que les données considérées comme confidentielles ne soient accessibles qu'à son propriétaire.
3. Garantie de disponibilité : s'assurer que les traitements puissent avoir lieu dans les délais prévus, avec les moyens prévus.

(RAMAEKERS 1984)

Le problème de la sécurité devient de plus en plus aigu au fur et à mesure que l'exploitation s'intensifie et qu'augmente l'importance stratégique des informations traitées. La première réaction d'un administrateur de système est alors de renforcer les contrôles, de complexifier les procédures d'accès, etc. Mais si la sécurité informatique est une exigence vu que l'information est un des actifs d'une société, la souplesse d'utilisation du système est un objectif tout aussi important (MINGER 1986). Il semble en effet nécessaire de considérer conjointement sécurité et performance : "...les contrôles nécessaires demandent des investissements et alourdissent les procédures de traitement. Cependant, les mesures de sécurité concourent en général à l'établissemnt de la performance globale du système informatique (RAMAEKERS 1984)".

Qualité de service et rigueur

Suite à leur expérience sur des micro-ordinateurs, les utilisateurs sont devenus de plus en plus exigeants. Ils attendent de ce "système central qui leur échappe", une disponibilité, une souplesse d'utilisation, des performances et des garanties de sécurité semblables. "L'obligation de délivrer un service de bonne qualité conduit à travailler dans une extrême rigueur pour assurer la parfaite fiabilité de tous les éléments entrant dans le processus (MINGER 1986)". Si les critères de qualité en matière de développement de logiciels ont fait l'objet de nombreuses études et publications, Elie note une

littérature nettement moins abondante en matière d'exploitation (ELIE 1983). Il propose de définir la qualité d'exploitation en terme de respect des budgets, satisfaction des utilisateurs et prévision d'évolution de la configuration. Nous pouvons facilement justifier les deux premiers critères si l'on considère que l'informatique est avant tout un service et que les bénéficiaires de celle-ci en sont aussi les commanditaires. Le troisième critère découle des deux premiers. D'une part, il est essentiel pour assurer la satisfaction des utilisateurs de pouvoir toujours faire face à la charge de travail imposé. D'autre part une grande rigueur s'impose dans la façon de prévoir l'évolution des besoins. Mieux les besoins seront planifiés, plus ils s'intègreront harmonieusement dans les budgets. Une acquisition de matériel non ou mal planifiée entraîne toujours un surcoût non négligeable dû, notamment, au délai demandé.

2.4 Conclusion : la nécessité d'une automatisation

Tout au long de ce chapitre, nous avons relevé un certain nombre de problèmes qui se posent depuis plusieurs années aux centres informatiques et qui dont l'acuité ne fera qu'augmenter dans les années à venir, surtout dans la partie exploitation de ces centres : les centres de calcul.

Après avoir approché les fonctions d'opérateur et d'administrateur de système dans leur complexité, nous avons identifié une évolution au sein des C.I. Celle-ci se manifeste par des changements au niveau du matériel, du logiciel, des applications et des rapports avec les utilisateurs. Ces changements ne manquent pas de poser des problèmes d'organisation majeurs aux C.I., complexifiant encore leur mission.

Au cours de notre développement, nous avons éludé toute forme d'automatisation, ignorant la présence d'outils logiciels susceptibles d'aider les exploitants dans leurs tâches. De tels outils existent bien entendu depuis plusieurs années, automatisant des aspects bien déterminés d'un système bien déterminé et survivant difficilement aux évolutions de sa configuration ou de sa politique opérationnelle. Beaver remar-

que combien les responsables des C.I. utilisent encore des méthodes de gestion antiques (BEAVER 1985) : ces responsables ont été tellement occupés à informatiser et automatiser les autres départements que l'on en arrive à donner raison au dicton "The shoemaker's children go without shoes". De nombreuses procédures exigeant beaucoup de travail sont encore manuelles. Non seulement ces procédures sont lentes mais elles sont de plus imprécises... il n'est pas possible de suivre avec précision l'utilisation des diverses ressources sans s'aider d'outils.

L'évolution que nous avons relatée amène progressivement les responsables à prendre la situation plus au sérieux et à considérer avec beaucoup moins de scepticisme les . promesses de l'automatisation. "Productivity is not to work harder... but work smarter !" déclarait-on à l'issue d'un colloque à Zurich : "Automating Machine Room Operations" (GEMIS 1985). Pour la plupart des gestionnaires, l'automatisation est devenue le seul moyen de faire face à la situation actuelle et de préparer l'évolution future.

On attend d'une automatisation qu'elle règle, du moins en partie, les problèmes évoqués plus haut, soit (1) qu'elle simplifie la tâche des opérateurs et administrateurs de système (dans ce dernier cas, le terme "assistance" paraît plus approprié qu' "automatisation") et améliore par la même occasion leurs conditions de travail; (2) qu'elle libère du personnel, au moins à temps partiel, afin qu'il puisse assurer les nouvelles fonctions d'assistance, de promotion et de conseil; (3) qu'elle prenne en compte l'inversion des rapports entre informaticiens et utilisateurs d'informatique, c'est-à-dire qu'elle offre une productivité et une qualité de service meilleure et (4), qu'elle aide à satisfaire les nouvelles exigences en matière de sécurité, souplesse d'utilisation et qualité de service.

Nous rediscuterons plus en détail l'intérêt d'une automatisation pour divers domaines clés de l'exploitation au chapitre suivant. Notons d'emblée que tous les objectifs d'une automatisation semblent se recouper : s'il semble nécessaire de

simplifier la tâche du personnel d'exploitation, ce n'est pas seulement afin d'améliorer ses conditions de travail... Diminuer l'intervention humaine permet en effet de travailler plus vite, à un coût plus bas et avec moins d'erreurs, satisfaisant ainsi les objectifs de performance, de productivité et de qualité de service.

Feldt note que l'idée à la base de toute volonté d'automatisation reste la performance (FELDT 1986) : les ordinateurs sont coûteux et il faut les faire "tourner" au mieux de leur capacité. Une automatisation s'accompagnant toujours de profonds changements dans l'organisation, en ce compris des réductions de personnel, il y a lieu de l'enrober dans un discours un peu moins terre à terre, mettant en valeur ses vertus sociales. Quand on se penche sur les revues d'informatique "business oriented", il est symptomatique de remarquer que plusieurs aspects présentés par les managers de C.I. comme des objectifs ne sont en fait que des effets de bord de l'automatisation. L'automatisation réduisant la participation du personnel d'exploitation simplifie fatalement sa tâche et améliore également ses conditions de travail. Quant à sa disponibilité pour assumer les "nouvelles fonctions", nous pouvons dire qu'il est heureux que de nouveaux besoins se soient développés au bon moment, simplifiant ainsi les problèmes de reclassement du personnel. D'autres effets de bord (ou objectifs ?) comme un contrôle accru du management sur les décisions de bas niveau sont moins clairement affirmés car plus difficilement avouables... on préférera parler de moindre dépendance de l'exploitation vis-à-vis d'un personnel spécifique.

L'automatisation n'est malgré tout pas la solution à tous les problèmes d'exploitation des C.I., de la même façon que l'informatique n'est pas la solution à tous les problèmes d'organisation des entreprises. La présidente d'une société commercialisant des outils d'aide souligne : utiliser un logiciel pour automatiser des procédures artisanales permet seulement de produire des erreurs plus rapidement qu'avant (BEAVER 1985). Le C.I. IBM d'Orléans conduit des recherches sur l'organisation et la mise en place d'automatismes (et non d'auto-

mates), afin de s'attaquer aux problèmes qui nous préoccupent. Cette approche alternative suppose une réorganisation afin de pouvoir constituer des entités ayant en charge des responsabilités bien définies et cohérentes. Ce processus repose sur des personnes clés dans l'organisation, appelées pilotes. Ces pilotes ont notamment comme mission de définir précisément toutes les activités impliquées dans le travail des membres d'une entité (MINGER 1986). Cette démarche n'est pas une réelle alternative à l'automatisation car elle ne peut résoudre certains problèmes "techniques". Cependant elle nous semble constituer une étape préalable à d'autres développements... il faut définir les tâches avant de les automatiser.

L'automatisation doit également s'accompagner d'une meilleure formation du personnel d'exploitation. Les compétences de celui-ci ne peuvent plus se limiter à des tâches très spécifiques qui tendent d'ailleurs à disparaître. Ce personnel doit être polyvalent au niveau de l'exploitation, rompu aux techniques traditionnelles et ouvert aux nouvelles technologies qui sont appelées à modifier considérablement le mode d'exploitation des centres informatiques. Rien ne sert d'automatiser, il faut restructurer à point pourrait-on dire ! Le personnel d'exploitation doit être, dès le départ, intégré au processus d'automatisation si l'on veut éviter de cuisants échecs.

Accroissement de la productivité, optimisation des ressources, amélioration de la qualité des services supposent donc de nouvelles méthodes. Celles-ci passent par une large automatisation, la définition de nouveaux métiers et l'utilisation de nouveaux outils d'information et de synthèse sur le fonctionnement des équipements, leurs taux de charge et l'utilisation finale des ressources informatiques... le tout pouvant déboucher sur l'instauration d'une valorisation, d'une facturation des services rendus. C'est en ces termes que le consultant décrit le service d'exploitation de demain (HÉRIARD 1986).

CHAPITRE 3 : INTERET ET FORME D'UNE AUTOMATISATION POUR DIVERS DOMAINES CLES

3.1 Introduction

Nous avons conclu le chapitre précédent sur l'idée que, face à la complexité croissante de la mission des exploitations, l'automatisation était un élément essentiel de toute stratégie. Si nous avons laissé jusqu'ici au lecteur le soin de ressentir cette nécessité en le sensibilisant à toute une série de problèmes, nous allons maintenant préciser l'intérêt que revêt l'automatisation pour les centres informatiques.

Les divers domaines envisagés ici constituent une division quelque peu artificielle des tâches d'un centre de calcul, qui ne correspond généralement pas à la répartition des fonctions. Il ne faut donc pas considérer qu'il existe nécessairement un responsable par domaine. Cette division a été envisagée afin de refléter au mieux celle existant entre les outils que l'on trouve sur le marché ou... dans les centres de recherche.

Dans un premier temps (3.2 à 3.7) nous décrirons brièvement, pour chacun des domaines, les tâches impliquées, nous en mettrons les difficultés en évidence et nous en déduirons l'intérêt d'une automatisation. Nous donnerons alors les grandes tendances existant en matière d'outils d'automatisation sur le marché en réservant, dans la mesure du possible, une attention particulière aux systèmes experts. Nous invitons le lecteur désireux d'obtenir déjà à ce niveau quelques informations sur la technologie des systèmes experts, à lire le point 2 du chapitre 4.

Afin de ne pas encombrer le texte, c'est seulement dans un second temps (3.8) que nous donnerons un tableau récapitulatif des outils existants. Notre première idée était de réaliser un survol assez complet de l'état de l'automatisation tant au niveau expérimental qu'au niveau commercial. Malheureu-

sement, si les prototypes font généralement l'objet d'articles dans des revues scientifiques ou d'exposés lors de conférences, il n'en est pas de même des outils déjà commercialisés. Les seuls documents s'y rapportant sont souvent les manuels d'utilisation qui, quand ils sont trouvables, sont très techniques. Nombre d'outils développés par des grandes firmes pour leurs besoins propres ont également été gardés secrets car incorporant une expertise rare. Ces problèmes de documentation sur des produits largement cités dans la littérature rendent parfois difficile leur classification dans une catégorie particulière.

Tous les domaines envisagés se recoupent, sont interdépendants et tournent autour d'une idée centrale : garantir la performance (au sens large) du système. De plus, nombre d'informations sont partagées par ces domaines ou échangées entre eux. C'est pourquoi nous concluerons ce chapitre en présentant une architecture idéale intégrant divers types d'outils et permettant d'envisager le contrôle de la performance d'un système comme un processus global.

3.2 Le contrôle opérationnel

Nous avons déjà parlé de contrôle opérationnel au cours du chapitre précédent en décrivant la fonction d'un opérateur (allocation des ressources, monitoring, traitement des problèmes) et la difficulté inhérente à celle-ci.

Le contrôle opérationnel consiste à s'assurer que le système remplit ses fonctions et atteint le niveau de performance attendu. Pour ceci, il s'agit d'analyser les messages apparaissant à la console et d'extraire des informations sur le système de façon continue ou occasionnelle. Quant un problème survient en terme de fonctionnalités ou de performance, celui-ci doit être rapidement identifié. Cette tâche d'identification passe souvent par l'analyse d'un fichier de "trouble tickets" qui regroupe l'expérience gagnée progressivement en la matière.

Dans le cadre du contrôle opérationnel, le temps est un facteur critique. Le temps dont dispose un opérateur pour résoudre un problème ou exécuter des procédures est très serré et varie du temps réel (réponse immédiate) à un jour. Tout délai dans l'exécution des actions a un impact négatif sur le niveau de service négocié avec les "clients", détériore leur productivité et va ainsi à l'encontre des objectifs économiques de l'organisation.

L'allocation normale des ressources est de plus en plus largement automatisée dans les grands centres. Ceux-ci ne peuvent en effet se permettre de gérer la planification des travaux batch sur une feuille de papier, ni de dépendre pour celle-ci de la mémoire d'un opérateur qui risque un jour de quitter l'organisation ! Il arrive également assez fréquemment qu'un opérateur se référant à une liste imprimée des jobs en attente, en exécute certains deux fois et d'autres pas du tout (BEAVER 1985). Automatiser permet d'éviter ces problèmes ainsi que les dépassements d'échéances ou les sous-utilisations du système lors des changements d'équipe. Les gains retirés d'une automatisation peuvent être importants : 13 heures toutes les deux semaines pour un centre gérant la paie de 16 000 ouvriers, après que celui-ci ait installé un package de 39.000 $ appelé "Automated Data Center System" (ADC2) (BEAVER 1985). Les avantages retirés d'une automatisation sont d'autant plus importants que celle-ci s'accompagne généralement d'une réduction de personnel.

En ce qui concerne le monitoring, de nombreux outils existent et sont couramment utilisés. Il s'agit de "moniteurs" qui tournent au niveau du système et demandent à l'ordinateur d'être son propre chien de garde en lui faisant collecter des données brutes à son sujet. Ces moniteurs mesurent la charge du système ainsi que la performance du CPU, des canaux d'E/S, de la mémoire et des périphériques. Ils peuvent être hardware comme Testdata, ou software comme SMS de Boole and Babbage. Parmi les moniteurs software, certains sont intégrés au système d'exploitation comme SMF Data sous OS/VS/MVS (cfr. 4.4.3). d'autres, comme BARS (pour les systèmes IBM), présentent les

données sous forme graphique.

Il existe très peu de produits automatisant la phase de traitement des problèmes de performance observés au cours du monitoring. Deux raisons semblent justifier cet état de fait, (1) il existe toute une série de problèmes assez rares, quoique possibles, qui sont très difficiles à détecter, diagnostiquer et résoudre et (2) les variations de politique opérationnelle d'un centre à l'autre rendent difficile l'élaboration d'un seul schéma de réponse aux problèmes opérationnels (MILLIKEN 1986). YES/MVS, un système expert expérimental que nous décrirons en 4.4 tente d'attaquer ce problème. A un échelon beaucoup plus modeste, EXPLORE/VM (Goal Systems International - Columbus, Ohio), permet de résoudre automatiquement certains problèmes en déclenchant des procédures lorsque des "warning points" sont atteints. Tout comme un opérateur, il rassemble dans un premier temps les informations complémentaires, nécessaires pour traiter le problème.

Il est malheureux que très peu d'outils soient disponibles à ce niveau car il existe un important marché en perspective. De nombreux centres aimeraient en effet s'assurer un "smooth running" sans pour autant faire les frais d'un système à tolérance de panne.

3.3 L'administration du système

Administrer un système consiste à assurer la stabilité des services offerts à court et moyen terme ainsi qu'à suivre le budget informatique afin d'éviter ou de justifier des dépassements budgétaires. Etant donné le niveau de service actuel, la configuration en place, les attentes des utilisateurs, la capacité et la charge prévue du système, l'administrateur se doit de négocier des objectifs en terme de niveau de service avec les utilisateurs. Quand les attentes des utilisateurs ne sont pas compatibles avec l'évolution prévue de la capacité du système, il demande d'abord une analyse de performance. Quand les actions de tuning prises suite à cela sont insuffisantes, l'administrateur doit alors demander une augmentation du budget. La tâche d'administration du système dépasse

le cadre du planning de capacité (cfr. 3.8) même si les liens sont très importants, car l'administrateur doit en plus veiller au respect des objectifs.

La difficulté majeure rencontrée ici provient de la tendance actuelle à limiter les ressources des départements informatiques. L'informatique a beau être de plus en plus stratégique pour l'entreprise, elle n'échappe pas aux mesures qui touchent tout département en période de récession. Or, une enquête réalisée auprès de responsables informatiques de grandes entreprises et présentée dans "Le Monde Informatique" (LMI 1986), montre que si la plupart des directeurs informatique se déclarent officiellement "décisionnaires" dans les investissements (80%) et dans la répartition du budget (81%), officieusement, ils reconnaissent que les décisions sont prises le plus souvent par la direction générale à la demande d'un service. Les directeurs informatique ont une étiquette de technicien dont ils ont du mal à se débarasser quand ils voudraient agir en stratège... ce qui veut dire qu'ils sont rarement décisionnaires au plus haut niveau et qu'ils doivent appuyer toutes leurs demandes avec des arguments solides, basés sur des prévisions précises. Souvent l'administrateur d'un centre passe le plus clair de son temps à prouver à ceux qui détiennent les cordons de la bourse qu'il est nécessaire d'acheter du matériel supplémentaire, de remplacer le matériel actuel ou d'engager davantage de personnel (BEAVER 1987). Ces discussions qui devraient idéalement avoir lieu lors de la détermination du budget, interviennent fréquemment devant des situations critiques ou pour justifier un dépassement de budget.

Un administrateur ne peut raisonablement pas obtenir les informations nécessaires (utilisation des ressources, évolution de la charge et du temps de réponse, etc...) sans outils. Ceci est d'autant plus vrai quand il s'agit d'un nouveau système en plein développement... il n'est alors plus du tout possible de se fier à son expérience ou à son feeling (BEAVER 1987). Les outils existants sont de divers types : (1) outils de monitoring présentant les données sous forme graphique, (2) des progiciels développés par des SSCI permettant d'utiliser de manière plus sophistiquée les fichiers obtenus des moniteurs

(corrélations, analyses statistiques, simulations de charge,...) et (3) des logiciels développés spécifiquement par une entreprise pour suivre certains critères. D'une manière assez générale, les directions informatiques utilisent peu ces outils, comme l'a montré une enquête réalisée par la SOCOTEC dans dix des plus grandes entreprises françaises (ELIE 1983). Lorsqu'elles les utilisent, elles ne présentent pas forcément leurs conclusions sous forme de tableaux de bord graphiques à l'usage d'une direction générale que l'on estime souvent peu intéressée par des détails. Celle-ci a malgré tout de plus en plus tendance à s'intéresser aux aspects techniques dans la mesure où ceux-ci se traduisent en définitive par des investissements en matériel, logiciel ou personnel.

3.4 L'interprétation des données

Une des tâches les plus couramment réalisées lors d'une analyse de performance est l'observation des données collectées par le système sur lui-même. Les moniteurs fournissant des informations en trop grand nombre, on utilise alors des outils de 'data reduction' et des instruments d'analyse qui traitent et réduisent les fichiers de données. Certains produits améliorent seulement la présentation ou offrent des possibilités de statistiques. D'autres permettent d'accéder aux données via un langage de haut niveau ou un système de menus; cette approche "base de données" est de plus en plus fréquente.

Les vendeurs de mainframes fournissent généralement à leurs clients de tels outils, mais ceux-ci sont souvent peu flexibles et leurs fonctionnalités limitées. Si de nombreux outils se valent quant à leurs fonctions, il se distinguent malgré tout au niveau de leur capacité à intégrer de multiples sources d'informations (données collectées par divers moniteurs ou dérivées par d'autres outils).

Le problème majeur rencontré ici provient du fait que tous ces outils fournissent malgré tout beaucoup trop de données sur la performance du système alors que l'analyste ne dispose que d'un temps très limité pour les interpréter et suggérer les mesures à prendre pour améliorer la performance.

L'intelligence artificielle pourrait aider ici à automatiser au moins le processus d'interprétation, en introduisant des règles pour prendre en compte les problèmes connus. REVEAL est un système expert expérimental destiné à améliorer la productivité de l'analyste dans sa tâche d'interprétation. Un système expert permet ici à l'analyste de travailler sur des problèmes de plus haut niveau, en analysant seulement les recommandations qui lui sont fournies. L'analyste fournit ainsi un temps de réponse nettement meilleur et peut toujours demander au système expert une justification de ses recommandations, améliorant parfois sa propre expertise.
(TERPLAN 1987)

3.5 La gestion des troubles

Différents types de troubles peuvent affecter un système informatique, du terminal que l'on a oublié de connecter jusqu'au circuit défectueux. Etant donné la technicité du domaine, nous ne rentrerons pas ici dans les détails. La gestion des troubles suppose trois types de tâches : la détection d'un problème, le diagnostic des causes de celui-ci et sa résolution. On classifie généralement les problèmes par ordre d'importance, d'après l'impact que ceux-ci risquent d'avoir sur l'utilisateur final. Selon la gravité du problème, celui-ci est traité par l'équipe opérationnelle en place ou confié à une société de maintenance spécialisée. La gestion des troubles suppose un suivi de ceux-ci, depuis leur détection jusqu'à leur résolution : un problème considéré comme mineur à un instant donné peut devenir crucial et exiger une intervention immédiate. Une fois le problème résolu, les symptômes observés ainsi que les solutions apportées doivent être consignés par écrit afin de pouvoir réagir plus rapidement à l'avenir dans des situations analogues.

Le premier problème rencontré ici provient du fait que les troubles sont souvent détectés trop tard, quand ils deviennent évidents aux yeux de tous. Il n'est alors malheureusement plus possible d'envisager une solution transparente pour l'utilisateur final. Le second problème est le diagnostic des causes des troubles dans la mesure où elles peuvent être très

nombreuses dans un environnement de grande taille. Il faut bien sûr noter qu'un centre informatique n'est pas complètement isolé, et que chaque constructeur s'occupe de la maintenance des systèmes qu'il vend et offre des facilités comme le télédépannage.

Les constructeurs qui doivent assurer la maintenance des systèmes qu'ils vendent et assurer un support de qualité à leurs clients estiment qu'il y a de bonnes raisons économiques d'automatiser le processus de gestion des troubles (GUILFOYLE 1986 b). Les "gros" clients sont également intéressés par des outils de diagnostic "in-house" qui leur éviteraient la lenteur d'une intervention extérieure. La formation d'un expert en "fault finding" étant très difficile et les erreurs de diagnostic coûteuses, ils se tournent tout naturellement vers les systèmes experts.

Affirmer que la technologie des systèmes experts n'a pas encore été assez éprouvée pour l'appliquer à la détection des troubles n'est pas un argument pour Rob Milne : l'essentiel n'est pas de rendre compte de tout le processus mais d'améliorer l'existant. Pour lui, des simples outils de diagnostic suffiraient déjà à améliorer la situation actuelle (GUILFOYLE 1986 b). Le réel problème des systèmes de diagnostic n'est pas la technique mais bien l'adaptation à l'utilisateur, à l'environnement ainsi que l'interfaçage avec les détecteurs matériels qui font partie d'un système. On ne peut en effet envisager de demander à un technicien de lire sans arrêt les valeurs fournies par l'équipement et de les entrer dans le système de diagnostic. Un outil de diagnostic utile doit être capable de puiser des données dans une variété de sources, dût-il en fin de compte poser malgré tout des questions à l'utilisateur comme dans les approches conventionnelles. L'adaptation à l'utilisateur et à l'environnement suppose également des contraintes. Il n'est, par exemple, pas toujours envisageable qu'après avoir détecté un problème, l'ordinateur en notifie directement une société de maintenance via une "call-out utility". Il est en effet peu probable que les clients de TANDEM (armées et banques) acceptent que quelqu'un soit averti de la faiblesse de leur système avant qu'ils n'en aient été conscients eux-mêmes (GUILFOYLE 1986 b).

Pour les utilisateurs qui ne disposent pas d'outils de collecte automatique d'informations sur leur système, il existe une catégorie de systèmes experts à leur portée : les manuels automatisés, de plus en plus populaires et ne recquérant que peu d'expertise. La plupart des manuels de détection des problèmes sont composés à 90% de listes de pièces de rechange facilement encodables dans une base de données. Les 10% restants qui montrent comment identifier les problèmes sont alors transformés en un ensemble de règles. L'objectif de ces manuels automatisés est de couvrir les cas relativement simples (GUILFOYLE 1986 b).

3.6 La gestion de la performance

La gestion de la performance d'un système ne peut être approchée comme une tâche, c'est un processus, un dialogue constant entre les différents domaines envisagés dans ce chapitre.

La figure 3.1 présente un diagramme des flux d'informations échangées dans le cadre de la gestion de la performance. L'administrateur fixe des objectifs de performance à l'expert, en terme de temps de réponse, d'utilisation des ressources, etc. L'expert informe l'administrateur du système de l'évolution de la situation au moyen de rapports non techniques... le nombre de SVC (Supervisor calls) n'intéresse pas l'administrateur par exemple. Sont d'un intérêt pour lui, (1) les variables à propos desquelles un objectif a été défini (ex.: temps de réponse inférieur à 1 seconde), (2) les variables qui sont sous le contrôle du centre et (3) les variables qui dépendent de l'extérieur mais auxquelles on peut répondre (ex. : volume des utilisateurs) (PILCH 1984).

L'expert récolte des informations sur le système au moyen d'outils de type "moniteur" qui lui fournissent des données ponctuelles ou agrégées. A l'issue de son processus d'analyse, il effectue alors diverses actions de tuning ou recommande l'achat de nouveau matériel afin de satisfaire les objectifs fixés. Nous examinerons plus en détail le processus de tuning en 4.5.

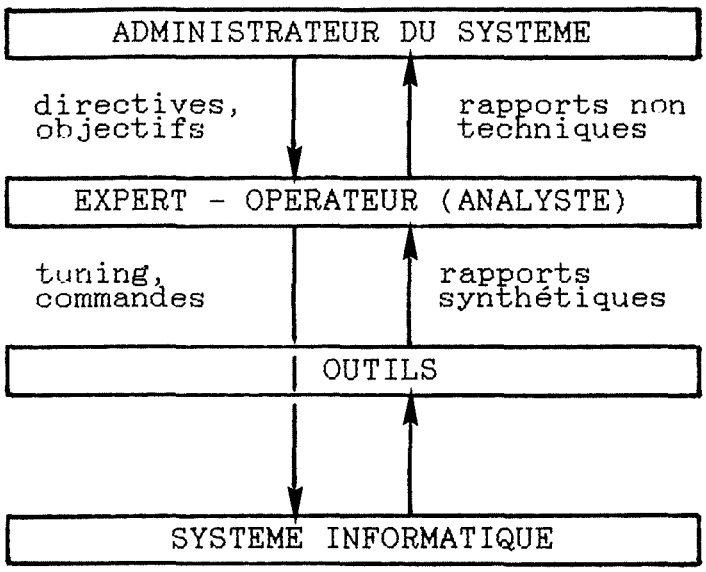

Figure 3.1 : Gestion de la performance

En ce qui concerne le processus d'analyse proprement dit, la méthode traditionnelle consistait à utiliser des procédés empiriques (rules of thumb) : lignes directrices observées sur des systèmes offrant un bon niveau de service et dans bien des cas, corroborés par des formules mathématiques. Par exemple, sur un système IBM 370 tournant sous MVS, un disque ne devait pas être occupé à plus de 30% et le CPU ne devait pas être utilisé à plus de 60% de sa capacité. La plupart de ces règles remontent à une époque où l'exploitation était essentiellement batch et où les utilisateurs pouvaient souffrir un délai mesurable en heures... un retard de cinq minutes dans l'exécution d'un travail était alors complètement transparent (HUNT 1986).

L'approche actuelle de l'analyse de performance est quelque peu différente. Le centre de calcul considère les objectifs de niveau de service pour déterminer si l'utilisateur reçoit ses travaux terminés dans les temps impartis, plutôt que de vérifier en permanence la performance du matériel. Quand une anomalie est détectée, on déclenche alors le processus d'analyse. La plupart du temps les requêtes proviennent de l'administration du système, du planning de capacité ou du contrôle opérationnel. Il convient alors d'accorder une attention particulière à :

- déterminer précisément les objectifs et les délais;
- régler le système en vue des exigences de service et de performance et non en vue de trouver la meilleure solution possible;
- évaluer à l'avance les avantages retirés;
- observer la règle des 80% / 20% (où 20% de la charge est responsable de 80% de la demande en ressources);
- considérer principalement les ressources critiques;
- déterminer le plafond de la capacité.

Une fois les hypothèses formulées, la faisabilité technique et économique doit être testée point par point afin d'exclure toute alternative non concevable. Après implémentation, des mesures doivent pouvoir vérifier et prouver l'amélioration de performance. Si l'amélioration n'est pas suffisante, de nouvelles hypothèses doivent êtres élaborées ou le planning de capacité revu (TERPLAN 1987).

Analyser la performance d'un système est un processus complexe qui recquiert une réelle expertise et suppose des relations très intenses avec l'administration du système, le planning de capacité et le contrôle opérationnel. Avant d'envisager "la" bonne action de tuning, d'importantes quantités de données doivent être interprétées et intégrées ou comparées aux résultats fournis par ces divers domaines.

Si jadis, pour se "couvrir", on pouvait se permettre une sous-utilisation des ressources en recourant à des méthodes empiriques, l'intensification de l'exploitation nécessite actuellement une gestion de la performance plus fine. Celle-ci supposant la manipulation de nombreuses données plaide en faveur d'une automatisation. De nombreux produits s'adressent en même temps au planning de capacité et à l'analyse de la performance, considérant ces domaines comme inextricablement liés. Parmi ces produits, les systèmes experts occupent une place honorable, du moins au niveau expérimental.

Outre les arguments classiques en faveur des systèmes experts comme la rareté des experts et la difficulté de leur formation, on retrouve ici les raisons suivantes.

- La performance des ordinateurs est importante pour la productivité des utilisateurs. Une étude a notamment montré qu'un mauvais temps de réponse accroissait l'anxiété de nombreux utilisateurs et par là même diminuait leur productivité (GUYNES 1988).
- Un système expert fournit une analyse de façon logique cohérente et reproductible.
- Les résultats peuvent être mémorisés dans des bases de données en vue de servir au planning de capacité par exemple.
- La connaissance des experts devient plus transparente.

Nous décrirons un tel système en 4.5. Les étapes de construction d'un système expert pour l'analyse de performance d'un système tournant sous VM sont données dans (GRALLA 1987).

3.7 Le planning de capacité

Comme nous l'avons souligné au point précédant, les liens sont très étroits entre planning de capacité et évaluation de la performance d'un système. Avant 1970, quand on s'intéressait surtout aux questions techniques de performance, ces deux termes étaient d'ailleurs utilisés l'un pour l'autre. Actuellement, "Computer Capacity planning", (CCP), est devenu un terme généralement accepté pour désigner les aspects de planning impliqués dans un centre informatique. Un autre terme est également employé par le management : "Computer Capacity Management", (CCM). CCP, CCM et CPE (Computer Performance Evaluation) sont trois termes employés dans le cadre du planning de capacité et ont des significations distinctes (CARPER 1983).

CPE (Computer Performance Evaluation) s'occupe des questions d'efficacité des programmes, des ressources hardware et de l'ordonnancement des travaux. Le tuning est une façon d'allonger la vie d'un système quand les limites de capacité sont atteintes, mais il ne prédit pas les futures exigences de capacité.

CCP (Computer Capcity Planning) a pour but d'assurer une capacité machine suffisante pour rencontrer les exigences de service des utilisateurs, et de prévoir la demande future en ressources informatiques, suffisamment tôt pour obtenir les éventuelles ressources additionnelles au prix le plus bas. CCP utilise des données sur l'existant (niveaux de service et d'utilisation, contraintes de la politique opérationnelle) combinées à des estimations de croissance et prédit la charge future et le point de saturation de la capacité.

CCM (Computer Capacity Management) reprend les informations de CCP et manipule l'utilisation et les niveaux de capacité du système en introduisant des contraintes artificielles. CCM peut par exemple facturer en-deçà ou au-delà du coût réel pour encourager ou décourager l'utilisation du système selon que sa charge est loin ou près de la saturation. Il peut également dégrader volontairement le temps de réponse afin que celui-ci soit semblable en période de faible et de forte utilisation. En bref, CCM constitue l'ensemble des contrôles du management qui affectent CCP. La figure 3.2 représente les relations hiérarchiques entre CCM, CCP et CPE. Plus on monte vers le sommet de la pyramide, plus les décisions sont stratégiques. Plus on descend, plus elles sont techniques.

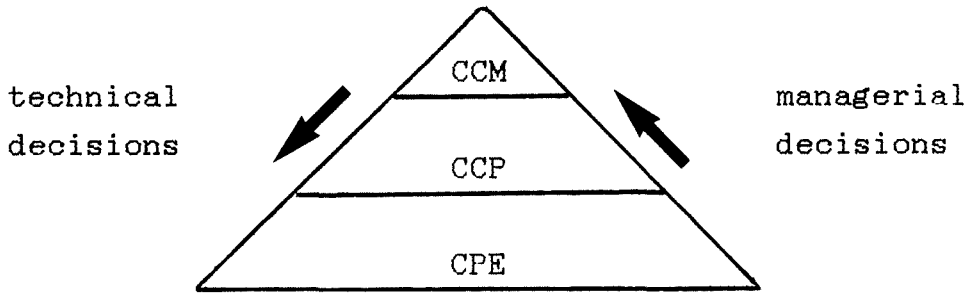

Figure 3.2 : Hiérarchie du planning de capacité
(CARPER 1983)

Cette vue hiérarchique du planning de capacité est très importante. Un processus de planning, aussi complexe soit-il, est inutile s'il n'est pas validé ou du moins porté à la connaissance du management. Les plans de capacité ne peuvent exister dans le vide, ils doivent prendre en compte les plans à long terme de l'organisation et s'y intégrer. Ce n'est donc pas au "low-level management" que l'équipe de CCP doit rendre compte... D'autre part, pour que le planning soit intéressant, il est essentiel que le management informe l'équipe de CCP des projets de développement de l'organisation.

En 1984, le planning de capacité ne semblait pas constituer une préoccupation majeure... Selon des enquêtes réalisées aux Etats-Unis et présentées dans (BRAUE 1984), 1 firme sur 5 a des problèmes sérieux de planning, 20 à 30% seulement essayent de planifier la capacité de leur système et 12% des "capacity planners" seulement font rapport au "top-management". Le directeur d'une firme spécialisée dans le CCP estime que 90% des achats d'équipement sont le résultat d'une pure conjecture, de la pression des vendeurs, de changements dans un arrangement de leasing ou de pressions soudaines pour accroître le service fourni (BRAUE 1984).

Les raisons sont multiples. Une raison culturelle est l'habitude qu'a le personnel d'exploitation de devoir régler des problèmes immédiats. Une seconde raison s'énonce en disant qu'il faut une fois faire face à un problème de saturation important avant d'être convaincu de la nécessité d'un planning (BRAUE 1984). Certaines organisations renoncent également à un planning précis tant il est difficile de prévoir les exigences des utilisateurs finals qui ont recours aux micros.

De plus en plus, le planning de capacité se présente comme une nécessité tant les économies en perspective sont importantes et les problèmes dûs à une saturation de la capacité graves... Un expert relate l'histoire de deux importantes compagnies de téléphone qui ont dû acheter des systèmes IBM semblables. L'une, n'ayant pas prévu cet achat à temps n'a pu négocier un rabais et a payé un excédent de 7 millions de

dollars par rapport à l'autre. A côté de celà, des organisations comme les banques qui entretiennent de nombreux contacts avec leurs clients via des ordinateurs ne peuvent se permettre des engorgements ou des délais à cause d'une saturation de la capacité machine (BRAUE 1984).

Des outils doivent être mis à la disposition des planificateurs afin de leur permettre d'intégrer les données en provenance du système (CPE), les données en provenance du marché des mainframes, les accords avec les utilisateurs, les accords avec les constructeurs et les objectifs définis par le management (CCM). La plupart des projets de développement s'étendant sur un minimum de cinq ans, des outils sont nécessaires tant pour simuler l'évolution de la charge que pour en réaliser le suivi.

De plus en plus de produits s'adressant au planning de capacité sont disponibles sur le marché. La firme GTE affirme avoir épargné 100 millions de dollars sur dix ans en planifiant soigneusement son augmentation de capacité. Le plan de GTE lui permet d'obtenir d'importants rabais en centralisant ses achats de matériel. Ce plan aurait été impraticable et prohibitif sans un ensemble de programmes et d'outils (moniteurs hardware et software) pour aider les "capacity planners" dans leur tâche. Le logiciel HRF (Hardware Resource Forecasting) comprend un ensemble de programmes qui consolident les données sur l'usage courant des ressources; fournissent des prévisions à cinq ans basées sur l'état actuel de l'utilisation, les grandes tendances futures et d'autres événements; fournissent des comparaisons entre la charge actuelle et les prévisions; et automatisent la sélection des ressources pour satisfaire les prévisions. Ces différents programmes sont installés autour d'une base de données : CUDB (Computer Usage Database). Un des inconvénients de ce logiciel vient du fait qu'il n'est pas directement basé sur le niveau de service (temps de réponse) mais bien sur la consommation des ressources. Les "développeurs" reconnaissent que, pour être basé sur le niveau de service, leur programme devrait posséder davantage de connaissances spécialisées sur, par exemple, la théorie des files d'attente (GTE 1984).

Les systèmes experts ne sont pas absents du marché des outils de planning, que du contraire. Il s'agit bien souvent, comme nous l'avons dit plus haut, de systèmes considérant ensemble le planning de capacité et l'évaluation de performance. Ces outils dérivent généralement dans un premier temps un modèle de l'ordinateur et de son exploitation en appliquant divers algorithmes aux données générées par les moniteurs de performance.

ISS three est un système expert pour l'évaluation de performance et le planning de capacité d'un système IBM sous MVS. La partie "capacity planner" est basée sur un PC connecté au mainframe qui fait en temps réel des recommandations de changement de configuration et fournit également des prévisions à long terme. Donald Russel de BGS Systems (Waltham, Mass.), vendeur de produits pour le planning de capacité se montre prudent quant aux promesses des systèmes experts dans cette branche (FELDT 1986). Cinq facteurs influencent selon lui la précision d'un planning :

- la qualité des données;
- la complétude des données;
- la qualité des algorithmes de modélisation;
- la complétude des algorithmes;
- la période d'observation de la performance.

Les systèmes experts ne peuvent remplacer de mauvaises données, de mauvais algorithmes ou une période trop courte. Là où ils peuvent être particulièrement utiles, c'est en cas de données incomplètes, par extrapolation. Par contre, utiliser la technologie des systèmes experts pour remplacer un manque d'algorithmes peut être dangereux. BEST/1 développé chez BGS est une alternative au système expert. Il dirige l'utilisateur à travers les différents pas d'analyse et attire son attention sur l'essentiel mais ne produit pas de conclusion sur la meilleure action à prendre.

3.8 Les produits existants

Le tableau 3.1 présente un grand nombre d'outils logiciels destinés à automatiser ou du moins faciliter les tâches qu'implique l'exploitation d'un centre informatique. Certains sont commercialisés, d'autres ont été développés pour les besoins internes d'une firme, d'autres encore existent seulement au niveau expérimental.

Ce tableau donne pour chaque produit, le nom de la firme qui l' a développé, ses domaines d'application, une référence pour obtenir davantage d'informations et, dans la mesure du possible, le type du produit (système expert, moniteur, etc). Nous l'avons construit sur base de nombreux articles issus principalement de revues "business oriented" mais également de revues scientifiques. Nous l'avons classé sur le nom de la firme qui développe le produit. CO, AS, ID, GP et PC sont des abréviations pour contrôle opérationnel, administration du système, interprétation des données, gestion des troubles, gestion de la performance et planning de la capacité.

PRODUITS	VENDEUR	DOMAINES						REFERENCE	TYPE
		CO	AS	ID	GT	GP	PC		
Tuning Aid	AT&T					*		(SAMADI 1987)	S.E.
NEMESYS	AT&T	*					*	(TERPLAN 1987)	S.E.
Mort	Automation Design Inc.		*					(BEAVER 1985)	Moniteur
NETADVISER	Avant-Garde	*	(*)		*			(TERPLAN 1987)	S.E.
DESIGNET	BBN						*	(TERPLAN 1987)	S.E.
SMART	Bell		*					(TERPLAN 1987)	S.E.
ACE	Bell		*		*			(TERPLAN 1987)	S.E.
TROUBLESHOOTER	Bell	*						(TERPLAN 1987)	S.E.
CAPTURE	BGS						*	(BRAUE 1984)	
BEST/1	BGS						*	(FELDT 1986)	(S.E.)
ESP	BGS					*	*	(TERPLAN 1987)	S.E.
CMF, IMF	Boole and Babbage						*	(BRAUE 1984)	Moniteurs
SMS	Boole and Babbage	*				*		(ELIE 1983)	Moniteur

Tableau 3.1 : Les produits existants.

PRODUITS	VENDEUR	DOMAINES						REFERENCE	TYPE
		CO	AS	ID	GT	GP	PC		
PERFORMANCE ADVISOR	Boole and Babbage			*		*	*	(TERPLAN 1987)	S.E.
BARS	Burroughs	*				*		(FELDT 1986)	Moniteur
Netman	California Software						*	(BRAUE 1984)	
ADC2	Cambridge Systems Group	*						(BEAVER 1985)	
Omegamon	Candle						*	(BRAUE 1984)	
X-ALERT	Cisi-Telematique	*			*			(DUBOIS 1986)	
CA-Jasper	Computer Assoc. Int'l						*	(BRAUE 1984)	
Torch PMS	Datametrics Systems						*	(BRAUE 1984)	
SPEAR	DEC				*			(PAU 1986)	S.E.
CDx	DEC			*				(BUCHANAN 1986)	S.E.
IDT	DEC			*	*			(PAU 1986)	S.E.
NTC	DEC	*	*		*			(BUCHANAN 1986)	S.E.
QCM	Duquesne Syst.						*	(BRAUE 1984)	
SRF	Gejac Inc.						*	(BRAUE 1984)	
ARSAP	Gejac Inc.		*				*	(BEAVER 1985)	
COSPER/D	General Motors					*		(TERPLAN 1987)	S.E.
TIMM-Tuner	General Research			*		*		(FELDT 1986)	S.E.
EXPLORE	GSI	*						(FELDT 1986)	Moniteur
COMPASS	GTE		*				*	(TERPLAN 1987)	S.E.
CICS, SMF, RMF	IBM					*	*	(BRAUE 1984)	
DART	IBM				*			(PAU 1986)	S.E.
RT PC	IBM			*				(GUILFOYLE 86b)	S.E.
YES/MVS	IBM	*						(ENNIS 1986)	S.E.
PROP	IBM	*						(WALMSLEY 1984)	S.E.
PINE	IBM	*			*			(TERPLAN 1987)	S.E.
RESLIM	IBM	*	*			*		(CHESS 1981)	Moniteur
CRIB	ICL			*	*			(PAU 1986)	S.E.
ARBY/NDS	ITT				*			(PAU 1986)	S.E.
Jars	Johnson Syst.						*	(BRAUE 1984)	
System Accounting	Macro 4 Inc.						*	(BRAUE 1984)	
Dumpmaster	Macro 4 Inc.				*			(BEAVER 1985)	
IMS/Maps	McAuto						*	(BRAUE 1984)	

Tableau 3.1 : Les produits existants (suite)

PRODUITS	VENDEUR	DOMAINES						REFERENCE	TYPE
		CO	AS	ID	GT	GP	PC		
MICS, TSO/MON	Morino						*	(BRAUE 1984)	
REVEAL	Morino			*				(TERPLAN 1987)	S.E.
Job Accounting/38	New Generation Software		*				*	(BEAVER 1985)	
Faultfinder	Nixdorf				*			(BUCHANAN 1986)	S.E.
Kommand	Pace Applied Tech.						*	(BRAUE 1984)	
Strobe	Programmart						*	(BRAUE 1984)	
Rabbit	Raxco						*	(BRAUE 1984)	
SAS	SAS Institute						*	(BRAUE 1984)	
Fast DASD	Soft. Corp. of America						*	(BRAUE 1984)	
SPSS-X	SPSS						*	(BRAUE 1984)	
Vsum	Star						*	(BRAUE 1984)	
Perform	Systems Applic.						*	(BRAUE 1984)	
DIAG8100	Travelers Insurance				*			(BUCHANAN 1986)	S.E.
ISS THREE	UCCEL					*	*	(TERPLAN 1987)	S.E.
U/ACR	Unitech Syst.						*	(BRAUE 1984)	
UCC-9/R+	University Computing Corp.						*	(BRAUE 1984)	
UCC-7	University Computing Corp.	*						(BEAVER 1985)	
DCMS	Value Comp. Inc.	*						(BEAVER 1985)	
VMAccount	VM Software						*	(BRAUE 1984)	
VMSchedule	VM Software						*	(BRAUE 1984)	
V/Safe, V/Snap	VM Systems Group	*			*			(BEAVER 1985)	
Main PM	Xidak						*	(BRAUE 1984)	

Tableau 3.1 : Les produits existants (suite et fin)

3.9 Vers une intégration des outils

Jusqu'ici nous avons pu observer à quel point les tâches d'un centre de calcul étaient interdépendantes :

- elles utilisent des informations communes sur le système, souvent issues d'outils de mesures communs;
- elles supposent de nombreux échanges d'informations;
- elles contribuent à la réalisation d'un même objectif : la performance (au sens large) du système.

L'intégration était souvent réalisée jusqu'ici en confiant à une seule personne la responsabilité de la cohérence de l'ensemble. Suite à la complexité croissante des systèmes et à l'intégration de ceux-ci au sein d'un réseau de transmission, chacune de ces tâches est devenue une spécialité requérant de nombreuses connaissances. Terplan propose une architecture intégrée pour le contrôle de la performance d'un système (figure 3.3). Nous n'allons pas décrire tous les détails de cette architecture idéale car nombre de ceux-ci sortent du cadre de notre travail.

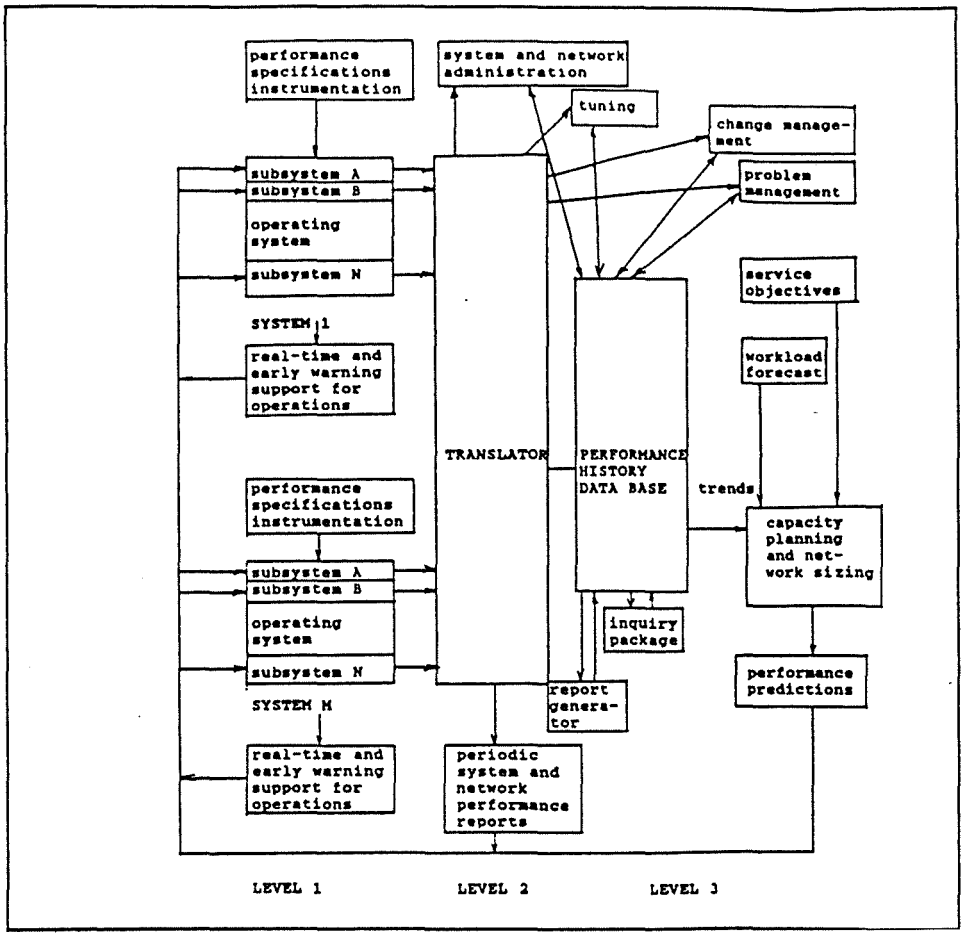

Figure 3.3 : Architecture idéale pour l'évaluation de la performance d'un système (TERPLAN 1987)

Au niveau 1, des moniteurs hardware ou software collectent des informations sur le niveau de service et l'utilisation des ressources. Un sous-ensemble de ces données agrégées est utilisé pour émettre des avertissements et contrôler les opérations en temps réel.

Ces données sont distribuées aux opérateurs et ont un cycle de vie prédéfini. Avant d'entrer dans la base de données de performance, les données sont transformées selon le format de la base. Le "translator" (niveau 2) peut être utilisé pour générer des rapports et interpréter les données. La base de données doit être plus qu'un ensemble de fichiers. Elle ne doit pas être trop gourmande en ressources, être capable de répondre à des requêtes complexes, offrir des possibilités de reporting et des techniques de compression des données. Cette base de données est utilisée par les diverses tâches dont nous avons parlé jusqu'ici, tant comme source d'informations commune que comme moyen d'archivage des résultats des analyses. Une base de données centrale règlerait donc à la fois les problèmes de redondances dans les données utilisées et les problèmes de communication.

Au niveau 3, on utilise les informations de la base de données afin de prédire les futurs niveaux de service et d'utilisation des ressources. D'autres informations sont également utilisées là, comme la charge et les nouvelles applications prévues, les niveaux de service attendus, ainsi que les alternatives technologiques envisageables. Les prédictions se font alors en utilisant des modèles.

CHAPITRE 4 : L'APPROCHE "SYSTEME EXPERT"

4.1 Introduction

Au cours du chapitre précédent, nous avons rencontré différents types d'outils que l'on peut classer selon leur capacité à prendre en charge les problèmes d'exploitation. Si l'on ne rentre pas trop dans les détails on peut considérer, des plus primitifs aux plus évolués, les trois types d'outils suivants.

Premièrement, des outils de type "moniteur", qui collectent des données sur le système. Ces outils, de conception relativement simple, n'ont pas besoin de la technologie analytique des systèmes experts.

Suite à la trop grande quantité d'informations fournies par ces moniteurs, nous avons également rencontré des outils qui automatisent en partie le processus de décision. Ceux-ci fixent des seuils de performance minimum et, une fois ces seuils atteints, déclenchent automatiquement des procédures de correction. Cette particularité est en-deçà des capacités des systèmes experts.

Enfin, nous avons cité de nombreux exemples de produits qui tentent de reproduire voire d'améliorer le savoir-faire des spécialistes de l'exploitation. Vu leur objet, ces outils utilisent tout naturellement la technologie des systèmes experts.

Si les vendeurs d'outils de monitoring de performance se tournent de plus en plus vers l'I.A., c'est parce que des outils capables de collecter des données et d'offrir des réponses prédéterminées face à certaines situations ne suffisent plus. Les exploitants ont de plus en plus besoin d'outils flexibles, offrant une réponse appropriée aux circonstances, et capables de faire des estimations adéquates face à de nouvelles situations... en quelques mots, des outils alliant la qualité, la flexibilité et le bon sens de l'expert à la vitesse de la machine.

La technologie des systèmes experts étant encore assez jeune, nous commencerons par une introduction à ses principes de base, précisant quelques termes "très AI" que nous utiliserons fréquemment par la suite.

On parle beaucoup actuellement d'une "mode des systèmes experts". Etant donné le coût associé au développement de tels produits, on ne peut se permettre de suivre une mode ! C'est pourquoi nous tâcherons, dans un second temps, de justifier l'approche "système expert" des problèmes d'exploitation.

Afin d'être plus concret, nous présenterons ensuite deux systèmes experts expérimentaux. Le premier, YES/MVS assiste, voire remplace, un opérateur moyen dans ses tâches non manuelles. Le second, TUNING AID, s'adresse plus spécifiquement au tuning d'un système d'exploitation UNIX.

Des nombreux avantages accordés aux systèmes experts, la plupart proviennent de leur mode de représentation des connaissances. Une représentation déclarative est-elle toujours préférable à une représentation procédurale dans les domaines qui nous préoccupent ? Nous discuterons cette question dans un troisième temps.

Une approche globale de l'automatisation des centres de calcul ne semblant pas très réaliste actuellement, nous présenterons dans un quatrième temps une technique qui pourrait être utilisée pour assurer l'intégration de divers systèmes experts spécialisés dans un domaine particulier de l'exploitation.

Nous terminerons ce chapitre par une évaluation de l'approche "systèmes experts", dégageant les forces et limitations de ces systèmes, les problèmes actuellement rencontrés dans leur développement et les perspectives qu'ils ouvrent dans les domaines qui nous ont préoccupés jusqu'ici.

4.2 Brève introduction aux systèmes experts, systèmes de production et "systèmes Shell"

Au cours des dix dernières années, un nouveau paradigme est apparu au sein de l'intelligence artificielle : celui de système expert. Ces systèmes sont - en théorie- capables de raisonner en suivant une démarche comparable à celle qu'adopte un spécialiste lorsqu'il résoud un problème relevant de sa discipline. Le concepteur d'un système expert doit avant tout recueillir l'expertise auprès du spécialiste (connaissances théoriques et empiriques). Cette phase d'acquisition des connaissances constitue déjà toute une entreprise dans la mesure où l'expert est à lui seul bien souvent incapable de formuler un savoir qu'il ressent comme "évident". Ce savoir consiste en un ensemble de méthodes assez mal définies que l'on appelle les heuristiques.

Les systèmes experts séparent systématiquement trois types de connaissances qui ne sont pas aussi nettement séparées dans un logiciel traditionnel :

La base de connaissances

La base de connaissances contient l'ensemble des informations spécifiques au domaine d'expertise. Elle est écrite dans un langage de représentation des connaissances où l'expert peut définir son propre vocabulaire. A l'inverse de ce qui se passe en programmation classique, l'ordre d'entrée des informations n'influe pas sur les résultats de sorte que chaque élément de connaissance est compréhensible par lui-même. La base de connaissances est parfois composée d'un ensemble de règles appelées "règles de production". Ces règles ont la forme "IF condition THEN action". La partie IF spécifie les conditions qui doivent être satisfaites pour que les actions spécifiées dans la partie THEN puissent être prises. On parle alors de systèmes de production.

La base de faits ou mémoire de travail

La "base de faits" contient les données propres au problème à traiter, mémorise les résultats intermédiaires et conserve une trace des raisonnements effectués. Elle peut donc être utilisée pour expliquer l'origine des informations déduites par le système.

Le moteur d'inférence

Le moteur d'inférence est un programme qui utilise les connaissances et heuristiques contenues dans la base de connaissances pour résoudre le problème spécifié par les données contenues dans la base des faits. Ce troisième type de connaissances est, idéalement, indépendant du domaine de connaissance de l'application. Il constitue une stratégie de contrôle pour le système, qui détermine la séquence d'application des règles de la base de connaissances. Il existe deux types de mécanismes d'inférence : le premier part des données disponibles et opère des déductions en se déplaçant de condition en action dans les règles de production. On parle alors de "data-driven reasoning" ou encore de "chaînage avant". Le second part d'une hypothèse et fait le chemin inverse. Il déduit des sous-hypothèses qui doivent être vérifiées pour que l'hypothèse de départ le soit et réduit ainsi le problème de départ en sous-problèmes dont la résolution finale ne dépend plus que de l'existence de faits.

La figure 4.1 donne un schéma très général de l'organisation d'un système expert. Moteur d'inférence, base de faits et base de connaissances constituent le système expert proprement dit. L'utilisateur communique avec celui-ci via un module d'interaction lui permettant, par exemple, d'émettre ses requêtes dans un langage proche du langage naturel. L'objectif de l'utilisateur est double : résoudre ses problèmes avec l'efficacité de l'expert et acquérir un savoir-faire analogue à celui de l'expert (vu que le système est capable d'expliquer son raisonnement). Derrière le module d'aide à l'acquisition des connaissances, se cachent des outils développés en vue de systématiser et de faciliter l'extraction des connaissances.

Ces outils ne remplacent cependant pas les spécialistes du transfert d'expertise dont le rôle est de faire "accoucher" les experts de leur savoir-faire.

La séparation des connaissances en trois entités comporte d'importants avantages. La codification du savoir des experts dans la base de connaissances est plus simple quand il ne faut pas se préoccuper des contrôles procéduraux. Le code résultant est plus facile à comprendre, à accroître et à modifier. Cette séparation permet également de créer des systèmes "shell" qui fournissent un langage de programmation avec des stratégies de contrôle incluses et qui peuvent être utilisés pour implémenter des systèmes experts dans une variété de domaines de connaissances. Pour davantage d'informations sur les systèmes experts, nous reportons le lecteur à GANASCIA (1985).

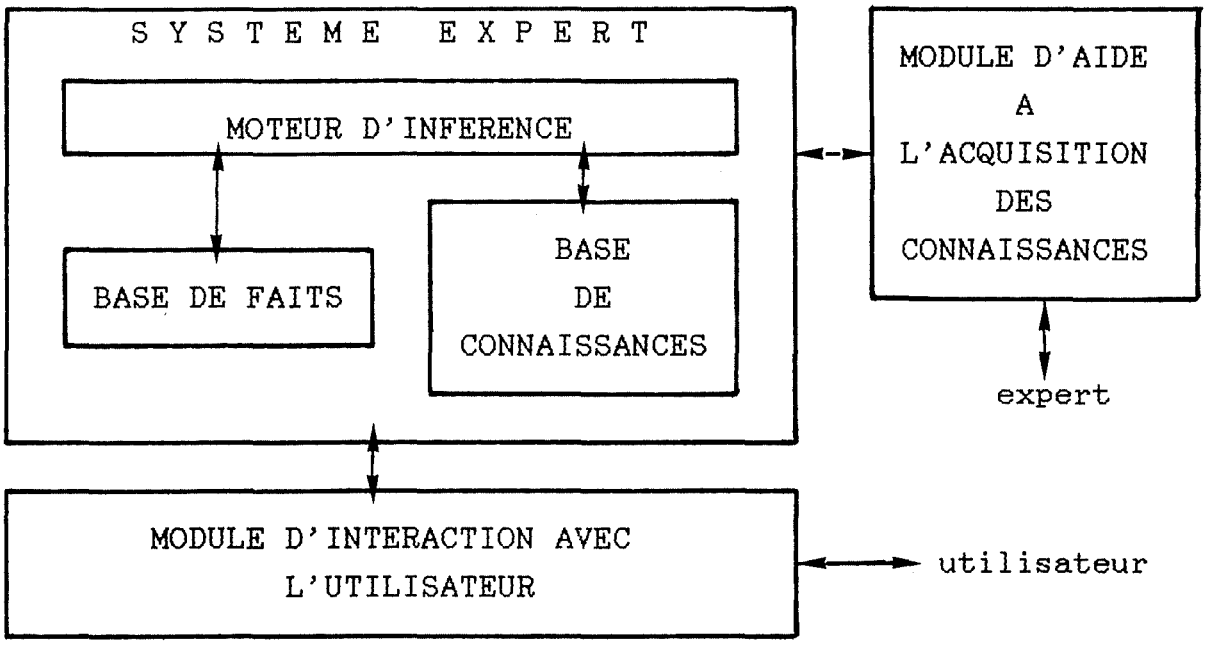

Figure 4.1 : Schéma de l'organisation générale d'un système expert

Bien que les sytèmes experts soient encore dans une phase de recherche, quelques réussites impressionnantes dans des domaines aussi variés que la médecine (Mycin), la géologie (Prospector) ou la configuration de systèmes informatiques (R1) sont à noter. Ceci amène à penser que les systèmes experts exerceront d'importantes influences sur l'évolution des sys-

tèmes informatiques : d'une part, ils enrichiront l'étendue des applications pour ces systèmes et d'autre part, ils fourniront de nouvelles techniques pour leur design, leurs opérations et leur maintenance.

4.3 Justification de l'approche "système expert"

Les systèmes experts, qui n'étaient encore qu'un sujet de recherches il y a quelques années d'ici, constituent actuellement un véritable marché. De plus en plus de sociétés au nom très "AI" se développent, proposant aux entreprises une approche "système expert" de leurs problèmes. Celles-ci hésitent encore à se lancer : les réalisations opérationnelles sont peu nombreuses et il est difficile d'évaluer leur intérêt véritable, les coûts de réalisation hors du cadre de la recherche sont encore mal connus, etc.

Même si les conditions concrètes de fabrication d'un tel produit commencent à peine à se dégager, on peut considérer actuellement le développement d'un système expert comme un processus long et coûteux. Sa construction suppose en effet la collaboration de nombreuses personnes hautement qualifiées durant plusieurs années parfois. Un tel coût ne peut se justifier que lorsque (TERPLAN 1987) :

- l'expertise humaine est rare;
- la formation de nouveaux experts est difficile;
- il existe une demande pour cette expertise;
- les problèmes sont bien définis et possèdent une solution;
- les problèmes sont complexes et il n'est pas aisé de les résoudre rapidement;
- les erreurs peuvent coûter cher.

Le tableau 4.1 évalue ces critères pour divers domaines. Le planning de capacité et le contrôle opérationnel semblent être des domaines dans lesquels le développement d'un système expert se justifie pleinement. C'est un système expert s'adressant au contrôle opérationnel que nous présenterons au point suivant; détaillons dès lors quelque peu la ligne du tableau qui s'y rapporte.

Domaines d'application	Critères						
	Expertise rare	Formation difficile	demande existe	Problèmes définis	Solution existe	Problèmes complexes	Erreurs coûteuses
Contrôle opérationnel	▮	▮	▮	▮	▮	▮	▮
Administration du système et du réseau	▮			▮	▮	▮	▮
Interprétation des données		▮	▮			▮	
Gestion des troubles		▮		▮	▮	▮	▮
Analyses de performance et tuning	▮	▮	▮			▮	
Calibrage du système et du réseau	▮		▮	▮	▮	▮	▮
Planing de capacité	▮	▮	▮	▮	▮	▮	▮

Tableau 4.1 : Justification des systèmes experts pour différents domaines clés (TERPLAN 1987)

L'expertise est rare et la formation des opérateurs difficile, comme nous l'avons déjà suggéré au point 2.2.3. Etant donné les variations de la configuration hardware, du software installé, de la charge et de la politique opérationnelle, un expert dans un centre ne devient pas facilement un expert dans un autre centre. Une longue période de formation est généralement nécessaire pour faire d'un opérateur nouvellement engagé un expert, ce qui représente un coût significatif pour l'exploitation. Le problème est aggravé par une pratique courante qui consiste à promouvoir les opérateurs expérimentés vers d'autres fonctions (analyste système par exemple).

Il existe une demande pour cette expertise, qui est fonction directe de la demande en informatique. Malgré la percée de la micro-informatique, cette demande est appelée à se développer considérablement dans les années à venir (cfr. 2.3).

Les problèmes sont bien définis et possèdent une solution. Il existe même parfois des manuels décrivant de façon formelle les procédures opérationnelles, manuels malheureusement trop volumineux et dès lors laissés pour compte au profit de solutions ad hoc expérimentées par les opérateurs. Notons ici que ces solutions artisanales qui font partie du "folklore opérationnel de l'installation" (MILLIKEN 1986) ne sont généralement pas distribuées ni documentées et dès lors ne sont pas appliquées de façon cohérente. Nous pouvons donc dire que si les solutions existent, il n'est pas aisé d'en obtenir une description formelle.

Les problèmes sont complexes, comme nous aurons l'occasion de nous en rendre compte au point suivant, et il n'est pas aisé de les résoudre rapidement à cause, notamment, de la quantité d'informations qu'il faut rassembler avant d'envisager une solution. L'environnement opérationnel étant caractérisé par une grande complexité et une évolution constante, il devient de plus en plus difficile de développer et surtout de maintenir des outils "procéduraux". Nous rediscuterons cet aspect au point 4.6.

Les erreurs sont coûteuses dans la mesure où les organisations dépendent de plus en plus de leur système d'information pour assurer leur bon fonctionnement. Toute erreur, même un simple retard ou oubli, peut avoir de fâcheuses conséquences, allant jusqu'à une paralysie momentanée de l'organisation.

4.4. YES/MVS, un système expert opérateur

4.4.1 Introduction

Le Yorktown Expert System/MVS Manager (YES/MVS) est un système expert expérimental qui assiste l'opération de grands complexes informatiques. YES/MVS fournit des conseils sur les opérations de routine, détecte, diagnostique et répond aux problèmes pouvant se poser à un opérateur. Une première version (YES/MVS I) a été utilisée régulièrement au centre de recherche IBM Thomas J. Watson dès 1984, un peu plus d'un an après le début du projet. Bien que YES/MVS I a prouvé la faisabilité du projet, sa viabilité économique restait à démontrer. C'est pourquoi une seconde version a été développée (YES/MVS II) en vue de faciliter l'adaptation du système à un site particulier, tenant compte de la configuration hardware, du software installé, de la charge et de la politique opérationnelle.

L'objectif d'une présentation détaillée de ce système est double : il s'agit d'une part de montrer la faisabilité d'un système envisageant l'automatisation des tâches opérationnelles de façon globale et, d'autre part, de révéler les problèmes sous-jacents à une telle approche. Nous allons donc tout

d'abord tenter de cerner le domaine d'application de YES/MVS I, de comprendre et d'évaluer ses choix d'implémentation, et de mettre en lumière les difficultés inhérentes au problème de l'automatisation des opérations d'un centre informatique. Nous tâcherons également de montrer à quel point YES/MVS (particulièrement la version II) constitue une expérience intéressante et une contribution non négligeable dans la résolution des problèmes que nous avons mis en lumière plus haut.

4.4.2 Présentation de YES/MVS I

Le Yorktown Expert System/MVS Manager (YES/MVS) est un système expert expérimental qui assiste en temps réel et de façon continue l'opération de grands complexes informatiques. Le système cible est un cluster d'ordinateurs IBM, chacun d'eux étant contrôlé par le système d'exploitation OS/VS2 MVS (Operating System Virtual Storage 2 Multiple Virtual Storage) ou plus simplement MVS. Les systèmes considérés ici fournissent tous des services à des utilisateurs interactifs via l'option Time Sharing de MVS (TSO) et à un ensemble de jobs soumis par ceux-ci, sous le contrôle de JES3 (Job Entry Subsystem 3 de MVS). Nous présenterons brièvement certains aspects de MVS dans la suite de notre exposé. Pour une information plus générale sur ce système d'exploitation nous reportons le lecteur à DEITEL (1984, pp. 567-600).

YES/MVS reçoit les messages qui apparaîtraient normalement à la console de l'opérateur, émet des requêtes et analyse les réponses en vue de connaître le statut du système. Suite à cette analyse, il soumet des commandes à MVS et à ses sous-systèmes. En résolvant les problèmes, YES/MVS affiche également des conseils pour l'opérateur et indique les tâches manuelles qui doivent être effectuées. YES/MVS peut être exécuté en mode "advisory" auquel cas l'opérateur doit confirmer les solutions proposées, ou en mode "active" auquel cas YES/MVS automatise complètement les aspects non manuels du travail de l'opérateur, avertissant ce dernier des décisions prises.

La base de connaissances de YES/MVS est composée d'un ensemble de règles de production. Celles-ci sont dérivées de l'expertise acquise par des opérateurs et programmeurs système et, dans une moins grande importance, de manuels du système. (ENNIS 1986 a)

4.4.3 Les domaines d'application de YES/MVS

Etant donné que la base de connaissances d'un système expert peut être aisément étendue, l'implémentation des diverses tâches d'un opérateur "morceau par morceau" est possible. Les domaines suivants ont été choisis pour une première implémentation dans la mesure où ils touchent à la majorité des activités d'un opérateur, (du moins celles qui ne requièrent pas d'interventions physiques) et fournissent ainsi un test suffisant des bénéfices attendus de la technique des systèmes experts.

Ordonnancement des grands travaux batch hors journée de travail

L'exécution des travaux batch importants doit être planifiée en tenant compte du "system throughput" et de la satisfaction des utilisateurs. Les aspects à considérer varient selon les installations. Nous détaillerons ce domaine en 4.4.6.

Management de SMF (System Measurement Facility)

SMF fournit des informations sur l'utilisation des données de MVS. Plusieurs actions de routine doivent être prises périodiquement (switch des buffers) ou à intervalles réguliers (exploitation des données pour la comptabilité).

Gestion de la taille de la file d'attente de JES

Avant, pendant et après leur exécution, tous les jobs lancés sous MVS sont conservés dans un fichier de spool central appelé "Job Entry Subsystem (JES) queue space". Les jobs sont éliminés de cette file d'attente une fois leurs outputs terminés (impression, transmission,...). L'opérateur est intéressé par l'espace disponible dans cette file d'attente car si celui-

ci est épuisé, JES ne peut plus rien faire (recouvrement impossible). Différentes actions doivent être prises dès qu'un seuil critique est atteint : par exemple, un dumping des gros travaux d'impression sur bande magnétique peut s'avérer intéressant... encore faut-il que ceci ait été prévu et qu'une bande ait été montée préalablement !

Problèmes dans les liaisons

Des ordinateurs installés sur un même site communiquent souvent via des liens de transmission "channel-to-channel". Si l'on ne maintient pas ces liaisons actives, le trafic des données est ralenti et l'espace disponible de JES peut être plus vite épuisé. Parmi les mesures à prendre :

- Examiner périodiquement l'état des liaisons afin de détecter d'éventuelles dégradations.
- Libérer les lignes des jobs à l'origine des troubles.
- Re-router les données via d'autres ordinateurs.

Erreurs hardware détectées par MVS

Quand MVS échoue dans le recouvrement d'une erreur hardware qu'il a détectée, il avertit l'opérateur. Même si celui-ci connait le moyen de résoudre ce problème (reconfiguration rapide, par exemple), il ne peut bien souvent pas répondre assez rapidement pour éviter un "crash". Les réponses aux problèmes hardware les plus fréquents ont été implémentées en règles indépendantes d'une configuration particulière. Les données décrivant la configuration hardware sont extraites de fichiers système de MVS, placées dans la base de faits et consultées par les règles.

Repos du système et chargement du programme initial

Dans certains cas comme l'installation de nouveau hardware, le test de nouvelles versions de software ou l'exécution de procédures de maintenance, il est nécessaire d'arrêter le système. On parle alors de fin de session planifiée. De multiples opérations sont nécessaires pour mettre le système au repos ("quiescing") et le redémarrer (Initial Program Loading -

IPL). Celles-ci nécessitent de nombreuses actions de l'opérateur et ralentissent considérablement le processus de reprise. Nous rejoignons ici un problème évoqué dans la première partie de ce travail : la "parameter file facility" que nous avions développée pour SLED avait également pour but de réduire l'intervalle de temps entre 2 sessions du système.

Monitoring de la performance

Surveiller la performance d'un système va au-delà de la tâche habituelle d'un opérateur. A court terme, il s'agit d'interpréter les données fournies par des moniteurs existants et, automatiquement, de détecter et de classifier les problèmes de performance en temps réel en générant des rapports sommaires. A plus long terme, il s'agit de diagnostiquer les problèmes de performance et d'essayer de les régler.

"Background Monitor"

L'activité principale de ce moniteur est de générer automatiquement des rapports sur les incidents survenus au cours d'une journée et de les communiquer via un courrier électronique aux programmeurs système responsables de sous-systèmes logiciels spécifiques.
(ENNIS 1986 b)

4.4.4 Le langage OPS5 étendu

Afin d'implémenter YES/MVS, OPS5, un langage pour systèmes de production a été choisi. OPS5 est un système SHELL (langage + mécanisme d'inférence) développé à l'université de Carnegie-Mellon. Ses avantages pour le projet YES/MVS sont les suivants :

- il est flexible et modifiable;
- il a pu être modifié afin d'être exécutable sous LISP/VM sur un ordinateur IBM;
- la représentation de la base de connaissances sous la forme de règles de production semble bien adaptée aux types de connaissances rencontrées dans le milieu des opérateurs;

- une forme d'inférence "data-driven" semble appropriée dans la mesure où il s'agit de répondre en temps réel à des informations reçues du système cible MVS.

Dans un système expert écrit en OPS5, nous retrouvons la mémoire de travail, la base de connaissances et le moteur d'inférence. Chaque élément de la mémoire de travail est membre d'une **classe**. Deux éléments d'une même classe possèdent la même liste d'**attributs** ainsi qu'une valeur pour chacun de ces attributs. La figure 4.2 reprend la définition de la classe d'éléments "printer-status". Toute élément de cette classe a des valeurs spécifiques associées à chacun des six attributs.

```
(literalize printer-status   ;Nom de la classe d'éléments de
                             ;la mémoire de travail
          reliable?          ;l'information est-elle fiable?
          address            ;adresse de l'imprimante
          forms              ;type de papier
          line-limit         ;nombre maximum de lignes
          current-job        ;numéro du job courant
          status)            ;autres informations...
```

Figure 4.2 : Définition de la classe d'éléments "printer-status" (ENNIS 1986 b).

Le second composant d'un système expert OPS5 est la base de connaissances. Celle-ci est composée d'un ensemble de règles de production. Une règle dont le membre de gauche est satisfait par un ensemble d'éléments de la mémoire de travail est appelée, avec la liste de ces éléments, instantiation de cette règle. A un moment donné, l'ensemble de toutes les instantiations de toutes les règles est appelé "conflict set".

La figure 4.3 donne un exemple de règle de production OPS5. Le symbole initial **p** identifie l'expression comme une règle. Le membre de gauche (précédant **-->**) contient trois éléments-condition qui doivent correspondre (matching) à des éléments de la base de faits pour que la règle soit instantiée. Chaque élément condition commence par le nom de sa classe ("task", par exemple) suivi d'un certain nombre d'attributs nécessaires au processus de matching ou de **liaison de variables**. Les éléments-conditions 2 et 3 offrent un exemple de liaison. Ils possèdent en effet tous deux un attribut ¬address

dont la valeur, donnée par la même variable <printer>, doit être identique.

Si les conditions sont satisfaites (correspondent à des éléments de la mémoire de travail à un jeu de substitution de variables près) et que le moteur d'inférence sélectionne cette règle, les actions spécifiées dans le membre de droite de la règle (suivant **-->**) sont exécutées. Il s'agit généralement de modifier (MODIFY), d'ajouter (MAKE) ou de supprimer (REMOVE) des éléments de la mémoire de travail. Les numéros apparaissant dans la partie droite de la règle font référence aux éléments de la partie gauche dans leur ordre d'apparition.

```
(p    jm:printer-status-update   ;nom de la règle
  (task                          ;identification de la tâche
    ¬task-id jm:info-collection);cfr. extensions à OPS5 infra
  (printer-status-reply          ;message de réponse de MVS
    ¬message-id iat8562          ;type message, printer-status
    ¬address <printer>           ;adresse de l'imprimante
    ¬status <status>             ;statut de l'imprimante
    ¬forms <forms>               ;type de papier imprimante
    ¬current-job <job>           ;job courant
    ¬line-limit <limit>          ;nombre maximum de lignes
  (printer-status                ;élt de la base de faits
                                 ;printer-status
    ¬address <printer>           ;adresse de l'imprimante
-->
  (remove 2)                     ;enlever la réponse
  (modify 3                      ;modifier l'élt de la base de
                                 ;faits "printer-status"
    ¬reliable? yes               ;marquer l'information fiable
    ¬status <status>             ;maj du status
    ¬forms <forms>               ;maj du format du papier
    ¬current-job <job>           ;maj du job courant
    ¬line-limit <limit>)         ;maj du nombre max. de lignes
```

Figure 4.3 : Règle de production OPS5 tirée du sous-domaine "JES queue" et utilisée pour maj les informations relatives à une imprimante quand de nouvelles données ont été reçues (ENNIS 1986 b).

Le troisième composant d'un système expert OPS5 est le moteur d'inférence. Le cycle d'inférence de OPS 5 est composé de trois phases (cfr. figure 4.4).

1. **Recognize** : déterminer l'ensemble des règles dont le membre de gauche est satisfait. OPS5 utilise le processus de matching "Rete".
2. **"Conflict Resolution"** : Si le "conflict set" contient plus d'une instantiation, ce qui est généralement le cas, OPS5 en sélectionne une en utilisant la stratégie de résolution de conflit LEX (lexical), qui est utilisée dans YES/MVS, ou MEA (means-ends analysis). Ces stratégies empêchent qu'une

instantiation soit exécutée plus d'une fois, favorisent les données créées le plus récemment dans la mémoire de travail et donnent la préférence aux règles ayant des parties 'IF' plus spécifiques.

3. **"Act"** : exécuter la règle choisie. Ceci entraîne des actions sur la mémoire de travail (MODIFY, REMOVE, ACT) ainsi que des appels à des fonctions LISP ou des routines MVS. Le système expert est stoppé une fois son "conflict set" vide.

(KASTNER 1986)

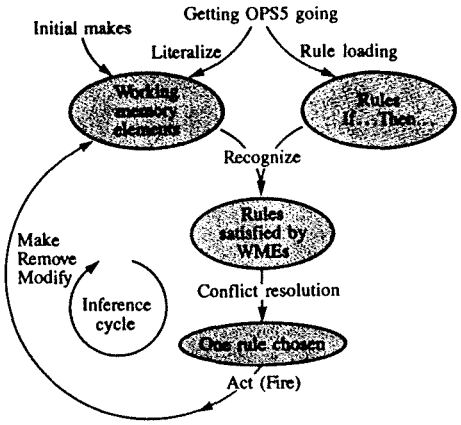

Figure 4.4 : Cycle d'inférence de base de OPS5 (ENNIS 1986 a)

Extensions à OPS5

YES/MVS est un système expert qui exerce un contrôle interactif, continu et en temps réel, c'est pourquoi plusieurs modifications de OPS5 ont du être envisagées (KARNAUGH 1985 & ENNIS 1986 a).

1. Il a fallu accroître sa vitesse d'exécution notamment en compilant la partie droite de chaque règle ou en répartissant les règles sur plusieurs systèmes OPS5 utilisant des processus concurrents (machines virtuelles - cfr. 4.4.5).

2. Afin de régler les problèmes de contrôle en temps réel une nouvelle primitive a été introduite, TIMED-MAKE, qui permet la production d'éléments dans la mémoire de travail après un certain délai.

3. Une autre exigence d'un traitement en temps réel est de permettre à des processus répartis d'interagir à des moments opportuns. Pour ce faire la primitive REMOTE-MAKE a été introduite. Celle-ci a les mêmes arguments que MAKE plus un attribut indiquant le destinataire de la production. De plus le cycle d'inférence de base a été modifié pour y introduire une phase de communication pendant laquelle les messages en provenance de l'extérieur sont collectés et les messages vers l'extérieur sont envoyés.

4. Il existe des problèmes critiques dans YES/MVS qui exigent qu'une séquence de commandes soit exécutée d'un trait, en priorité. Les deux modes de résolution de conflit (LEX, MEA), pouvant aller à l'encontre de ces objectifs, ont été étendus à un "Priority Mode" qui assure que seules les règles ayant la plus haute priorité sont soumises à la stratégie de résolution de OPS5. A chaque règle est ainsi associé un élément "task" (cfr. figure 4.3) et à chaque type de tâche est associée une priorité. On peut ainsi s'assurer qu'une tâche urgente ne sera pas interrompue par une tâche moins urgente mais plus récente !

5. YES/MVS ayant à assurer un contrôle continu, il a fallu implémenter en OPS5 la règle OPS-WAIT qui met le système en état d'attente quand le "conflict-set" est vide, plutôt que de le stopper. N'importe quel message externe (en ce compris un événement déclenché par un TIMED-MAKE) entraînera la reprise du système.

La figure 4.5 montre le cycle d'inférence modifié, tenant compte des exigences que nous venons de décrire.

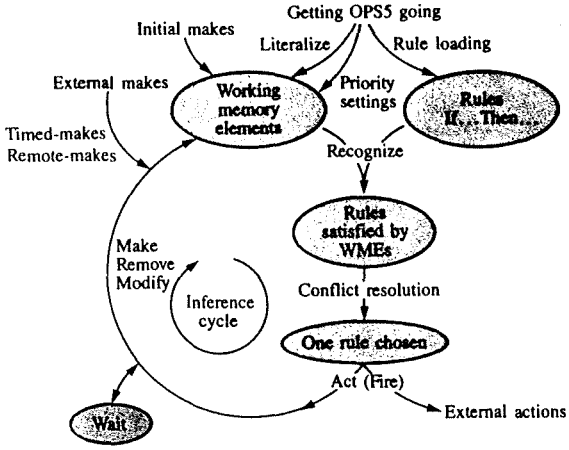

Figure 4.5 Le cycle d'inférence de YES/MVS (ENNIS 1986 a)

4.4.5 Organisation de YES/MVS

Afin d'être à même de traiter la plupart des incidents du système cible, YES/MVS est exécuté sur un ordinateur distinct (système VM/370) et n'interfère pas avec les opérations de MVS. De plus, si le système VM/370 est en difficulté, MVS peut continuer à opérer sous contrôle manuel. Le seul interface avec MVS est une émulation d'une console JES3 sur PC, apparaissant aux yeux de MVS comme une console classique, mais étant accessible en lecture et en écriture à YES/MVS.

YES/MVS est implémenté en trois machines virtuelles sous VM/370 : La machine virtuelle "**expert**", la machine virtuelle "**MVS Communication Control Facility (MCCF)**" et la machine virtuelle "**Display Control**". MCCF communique avec MVS via un système développé séparément : "**Centralized Computer Operation Project (CCOP)**". Le but de CCOP est de centraliser le contrôle et de filtrer les messages entre différents ordinateurs d'une installation et leurs opérateurs (cfr. figure 4.6). L'adoption de ce design pour YES/MVS a plusieurs avantages : une exécution asynchrone des différents processus qui composent YES/MVS permet d'obtenir de meilleures performances et le système expert est libéré de considérations I/O telles que le format des messages. Pour une description du système d'exploitation VM et du concept de machine virtuelle, nous reportons le lecteur à (Deitel 1984, pp. 601-629).

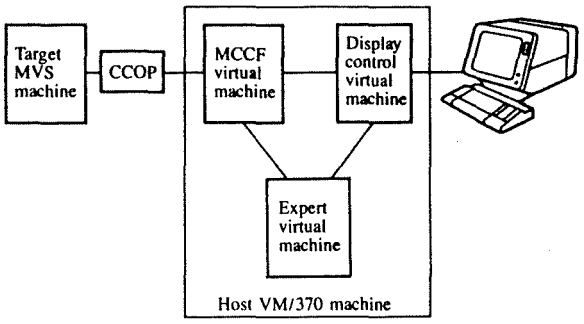

Figure 4.6 : Organisation de YES/MVS (ENNIS 1986)

Les machines virtuelles "expert" et "display control" sont toutes deux implémentées en OPS5 étendu et communiquent entre elles par le biais de messages ayant le format des éléments d'une mémoire de travail OPS5. Tout message reçu par l'expert devient une partie de sa mémoire de travail. MCCF traduit les messages passant dans les deux directions entre l'expert et MVS. Il filtre également les messages en provenance de MVS, sélectionnant ceux qui ont un intérêt pour le système expert. MCCF est implémenté en REXX (system exec language).

L'expert communique avec l'opérateur au moyen d'une hiérarchie d'écrans qu'il envoie à la console. Au niveau global, les messages apparaissent en une ligne et en différentes couleurs indiquant le type et le statut. L'opérateur peut également examiner des écrans détaillant l'action à prendre avec les commandes à utiliser et un texte explicatif. Si YES/MVS est exécuté en mode "advisory" et qu'une action est recommandée, il peut l'autoriser, l'exécuter lui-même ou la refuser. En mode "active", l'opérateur est averti des actions prises et peut également envoyer des commandes non sollicités par l'expert. Ceci peut être utile quand on veut temporairement supprimer certaines actions "normales" du système.
(KARNAUGH 1985)

4.4.6 Détail d'un sous-domaine : Ordonnancement des grands travaux batch hors journée de travail

Acquisition des connaissances

Afin de développer la partie de la base de connaissances pour ce sous-domaine, des membres de l'équipe opérationnelle et du management furent interviewés en vue de dégager les méthodes empiriques et les procédures qu'ils utilisaient.

```
(p remove-job-when-it-won't-finish
                                        ;SI
  (time ¬time-left <time-left>)         ; temps restant

  (job-order                            ; un job
    ¬state awaiting-execution           ;  attendant son exécution
    ¬in-q true                          ;  qui est dans la file
    ¬adj-cpu-time > <time-left>         ;  qui ne sera pas terminé
    ¬job-id <job-id> )                  ;  identifié par <job-id>

 -->                                    ;ALORS
  (make request                         ; La stratégie consiste à
    ¬do remove-job-from-queue           ; rectifier l'incohérence
    ¬job-id <job-id> ) )                ; à retirer le job
```

Figure 4.7 : Règle chargée de retirer de la file d'attente les jobs qui ne seront pas terminés avant la le changement de shift (SCHOR 1986)

1. Vu la division de l'exploitation en "shifts" (day shift, overnight batch shift), il faut éviter de programmer un job qui ne serait pas terminé avant le changement de shift (cfr figure 4.7), exécuter les jobs les plus longs en début de shift afin de s'assurer qu'ils soient terminés et les jobs les moins importants en fin de shift afin de combler au mieux le temps disponible.
2. Pour assurer une certaine égalité entre les utilisateurs, utiliser une stratégie d'ordonnancement "round-robin" : ne pas exécuter le Nième job de A tant que le N-1ième job de B n'a pas été exécuté.
3. Tenir compte des priorités : les jobs restants des jours précédants ont priorité sur les jobs du jour et d'autres priorités peuvent être accordées par l'opérateur.
4. L'heure de début et de fin de shift peut être changée dynamiquement. Une erreur hardware peut exiger que le système soit libéré plus tôt pour faire place à l'équipe de réparation.

5. Les nouveaux jobs sont dynamiquement ordonnancés à leur arrivée.
6. Normalement le nombre de jobs pouvant être exécutés à la fois est de quatre, mais ce nombre peut être changé par l'opérateur. Les jobs ayant le même nom sont éxécutés séquentiellement.

Organisation de la base de connaissances

On peut distinguer six groupes de règles.

1. Création et maintenance d'un modèle de MVS et des jobs en attente.
2. Règles d'ordonnancement
3. Règles de fixation de priorités
4. Transmission des résultats à l'opérateur et/ou MVS
5. Contrôle des "shifts" (in shift, not in shift, starting,...)
6. Autres.

La prise en compte des contradictions

Durant les opérations, un modèle cohérent de l'ordre idéal de la file d'attente des travaux est dérivé des informations (changeantes) de MVS. Nous avons donc un modèle d'un système en temps réel. Quand le système change au cours du temps, les faits représentant le système changent également, ce qui entraîne des contradictions entre les faits et les conséquences préalablement déduites des anciens faits. Retrouver un état cohérent correspond à implémenter une sorte de "Truth Maintenance System (TMS)". Un exemple de règle de "truth maintenance" est donné en figure 4.8. Il s'agit ici de maintenir les jobs à exécuter dans une file ordonnée. Chaque job a un pointeur vers le suivant. Chaque job a également un numéro correspondant à son ordre dans la file. Quand un nouveau job est inséré dans la file (ou un ancien retiré), les numéros d'ordre doivent être ajustés. Cette incohérence est détectée quand deux jobs qui se suivent dans la file ont des numéros qui ne sont pas séparés par 1. Quand un job est inséré, la règle est exécutée de façon répétitive pour retrouver un état cohérent. (ENNIS 1986 a)

```
(p tms:order-nbr-wrong                    ;nom de la règle
  (task                                   ;priorité de la tâche
    ¬task-id normal+1)
  (job-ordre                              ;un job
    ¬next-job-id <next-job-id>            ;  dont le job suivant est
                                          ; <next-job-id>
    ¬in-q t                               ; qui est dans la file
    ¬order-nbr+1 <order-nbr+1>)           ; ayant un <order-nbr+1>
{ <nxt-job>
  (job-order                              ;le job suivant
    ¬job-id <next-job-id>
    ¬in-q t                               ; dans la file
    ¬order-nbr <> <order-nbr+1>) };       ; dont le n° d'ordre n'est
                                          ; pas 1 + le n° d'ordre du
                                          ; job précédent

 -->
 (modify <nxt-job>                        ;la stratégie pour rectifier
    ¬order-nbr+1                          ;l'incohérence : corriger le
      (compute 1+<order-nbr+1>)           ;numéro d'ordre du job
    ¬order-nbr <order-nbr+1>))            ;suivant
```

Figure 4.8 : Règle de "truth maintenance" utilisée pour maintenir une liste ordonnée des jobs en attente (ENNIS 1986 a)

4.4.7 Problèmes rencontrés au cours du développement de YES/MVS

Acquisition des connaissances

La principale difficulté rencontrée à cet égard dans YES/MVS provient de l'objectif que ce système s'est fixé, à savoir, non seulement conseiller l'opérateur mais également automatiser ses fonctions (n'exigeant pas d'intervention manuelle). Vu les limitations d'un opérateur en terme de vitesse, de résistance et de formation, le système est potentiellement capable d'une activité plus importante. YES/MVS ne devant pas se borner à imiter l'opérateur, il est nécessaire d'acquérir d'autres connaissances, soit en demandant à l'opérateur ce qu'il ferait s'il en avait le temps et les moyens, soit en faisant appel aux connaissances des programmeurs système. (KARNAUGH 1985)

Méthodes de test

De nombreux problèmes ont été rencontrés à cet égard:

- la complexité de MVS rend prohibitif le développement d'un simulateur;
- des tests sur un système "réservé aux tests" ne rendent pas toute la complexité d'un environnement réel;

- certains tests nécessitant un sabotage du système cible entraînent une dégradation du service, même s'ils sont réalisés hors journée;
- même en effectuant les tests sur un environnement réel, la recréation de certaines situations particulières, en vue de tester l'effet d'une modification de la base de connaissances, est parfois très difficile;

(ENNIS 1986 a)

Nous pouvons ajouter que, dans une perspective d'utilisation de ce système dans d'autres centres, toute une série de tests devraient être à nouveau effectuée afin de tenir compte de la configuration hardware et de la politique opérationnelle en place. Il ne s'agit donc pas d'un mince problème...

4.4.8 Evaluation de YES/MVS I

La motivation du développement de YES/MVS I était de construire un système expert dans le cadre d'une application réelle afin d'appréhender l'intérêt de cette nouvelle technologie. Les conclusions tirées à l'issue de cette expérience sont positives.

YES/MVS a été utilisé de façon régulière durant une période de plus de neuf mois au centre de recherche de Yorktown, d'abord en mode "advisory" puis en mode "active". Selon les statistiques réalisées au cours de cette période, les machines virtuelles "expert" et "display control facility" ont utilisé en moyenne, respectivement 0.3% et 0.57% de la capacité du processeur central d'un IBM 3081-K. Les opérateurs ont accueilli ce projet favorablement estimant que YES/MVS les alertait au sujet de problèmes qu'ils ignoraient et les conseillait efficacement pour les résoudre (ENNIS 1986 b).

Les responsables du projet ont estimé, à l'issue de YES/MVS I qu'il pouvait être économique de développer un outil pour accroître la productivité des opérateurs ou la qualité de la fonction qu'ils offrent. Les coûts de développement assez élevés constituent pour eux des "coûts d'entrée " associés à toute nouvelle technologie et sont également dûs à la nécessité de créer des outils d'aide au développement - inexistants jusque là. Ces coûts leur semblent se justifier par le gain de productivité attendu des opérateurs, la préservation et la distribution de leur savoir ainsi que par les horizons ouverts... Notons tout de même que l'implémentation des deux sous-domaines (ordonnancement des travaux batch et gestion de la taille de la file d'attente de JES), ont nécessité la coopération de 8 "développeurs" et 1 expert durant plus de deux ans, à raison de 3 jours par semaine, sans compter les nombreuses aides et participations épisodiques (ENNIS 1986 b).

Certains problèmes inhérents restent cependant à attaquer :

- la complexité du domaine des opérations;
- les variations de politique opérationnelle d'un site à l'autre;
- l'évolution d'un système informatique et de la politique opérationnelle sur un même site;

(MILLIKEN 1986)

4.4.9 De YES/MVS I à YES/MVS II, un système Shell pour l'automatisation des opérations

Une leçon importante a été tirée du développement de YES/MVS I : l'automatisation des opérations de MVS dépasse le développement d'un unique système expert. On pourrait s'attendre à ce que les tâches opérationnelles d'un mainframe sous MVS soient relativement semblables à celles d'un autre mainframe sous MVS mais il n'en est rien. Selon Miliken, un des "développeurs", seulement 50% des résultats de YES/MVS I pourraient être réutilisés hors du site de Yorktown. Il existe en effet de nombreuses différences d'un site à un autre concernant la configuration et la politique opérationnelle en place : la responsabilité accordée aux opérateurs, par exemple, varie

grandement d'un centre à l'autre. C'est pourquoi, il serait nécessaire de développer une famille de systèmes experts.

YES/MVS II a pour but la création d'un système Shell qui serait à la base de l'automatisation des opérations pour une famille de systèmes experts. Les deux objectifs du projet sont (1) d'y incorporer les services communs à tout système expert opérateur MVS et (2) de faciliter l'adaptation d'un système expert pour un site particulier. Un telle approche améliorerait grandement la faisabilité économique de l'automatisation des opérations. Les trois sites sur lesquels le système est actuellement en utilisation - à Yorktown, Poughkeepsie et Kawasaki au Japon - continuent le développement de YES/MVS II, de façon telle que les "développeurs" puissent découvrir un "noyau commun" aussi important que posssible.
(GUILFOYLE 1986 a)

4.4.10 YES/L1 et la représentation des connaissances

Plusieurs raisons ont poussé les concepteurs de YES/MVS à développer un nouveau langage pour la version II. On aurait pu s'attendre à ce que ceux-ci jettent leur dévolu sur le flambant neuf ESE (Expert System Environment) développé par IBM mais la représentation des données en ESE ne correspondait pas du tout à celle favorisée dans YES/MVS. Le langage développé s'appelle YES/L1 (Yorktown Expert Systems/Language One), est 'data-driven', 'rule-based' et inclue un sous-ensemble de PL/I. Nous reportons le lecteur à (CRUISE 1987) pour une présentation détaillée de YES/L1. Nous nous bornons ici à donner quelques motivations à la base de son développement (insuffisances de OPS5).

1. YES/L1 permet de minimiser les interactions entre les règles rendant ainsi la base de connaissances plus lisible et plus aisément modifiable. A chaque règle correspond idéalement une "situation-réponse" de l'opérateur.
2. YES/L1 n'impose plus aux éléments de la mémoire de travail référencés dans la partie droite de la règle d'apparaître dans sa partie gauche. Les parties gauches correspondent dès lors davantage à des situations de l'opérateur.

3. Dans OPS5 un élément de la mémoire de travail est considéré comme nouveau dès que l'un de ses attributs a été modifié. Il peut dès lors être à nouveau instantié et ceci peut induire des effets indésirables... considérons par exemple une règle "list-3211-printers" dont le but est d'avoir une liste de toutes les imprimantes IBM 3211. Cette règle ne s'intéresse donc qu'à l'attribut "¬type" des éléments "printer". Si un changement est fait à l'attribut "¬status" d'un élément "printer", un nouvel élément est créé qui pourrait déclencher la règle. Cet effet n'est ici pas désirable. Avec YES/L1, une instantiation ne rentre à nouveau dans le "conflict-set" qu'en cas de modification d'un attribut référencé par la règle.
4. Dans YES/L1, le membre droit d'une règle peut employer n'importe quelle construction PL/I. Les actions réalisées par un opérateur s'expriment en effet souvent de façon procédurale.

(CRUISE 1987, MILLIKEN 1986)

4.4.11 Management d'un modèle de l'environnement

Gérer un modèle du statut de l'environnement extérieur (système cible) est un élément clé dans un système expert comme YES/MVS. L'absence d'une telle gestion dans YES/MVS I a amené plusieurs problèmes.

1. Redondance dans les requêtes effectuées par les experts des différents sous-domaines de YES/MVS.
2. L'information (relative au système cible) se trouvant dans différentes classes de la mémoire de travail, il est difficile de la partager. De plus, l'information étant dupliquée, des incohérences sont possibles.
3. Il n'y a pas de contrôle central sur la politique d'émission de requêtes (combien de fois le système cible doit-il être interrogé ?, ...)
4. Quand des requêtes concernant le système ou l'opérateur, peuvent être émises dans différents sous-domaines, il est difficile de gérer des priorités entre celles-ci, ce qui serait pourtant intéressant dans des situations complexes où un problème urgent doit être résolu.

Le "model manager" de YES/MVS II répond à ces problèmes en définissant une représentation standard pour le statut du système-cible et en maintenant en permanence un modèle de celui-ci en mémoire de travail.
(MILLIKEN 1986)

4.5 TUNING AID, un système à base de connaissances pour le tuning d'un système d'exploitation UNIX

4.5.1 Introduction

"Tuning Aid" est un premier résultat d'un projet de recherche en cours chez AT&T Bell Laboratories, chargé d'évaluer l'intérêt d'automatiser le processus de tuning d'un système d'exploitation UNIX. Le tuning d'un système, dont le but est souvent d'améliorer sa performance, implique diverses activités :

- ajuster certains paramètres du système comme le nombre de buffers;
- exécuter des routines de maintenance comme "dcopy" chargé de réorganiser un système de fichiers sur disque;
- développer des règles opérationnelles comme l'exécution de backups hors des heures de pointe;
- acheter du nouveau hardware,...

Le tuning d'un système

Le tuning d'un système d'exploitation se fait en trois étapes. Premièrement, les mesures doivent être interprétées en vue de découvrir des problèmes de performances ou des possibilités d'amélioration. Il s'agit d'une tâche fastidieuse vu la quantité de données à traiter. Deuxièmement, une action de tuning appropriée au problème détecté doit être prise. La détection d'un problème et le choix d'une solution exigent une bonne maîtrise des aspects internes et opérationnels du système. Troisièmement, les conséquences de toute action doivent être évaluées, soit a priori, en utilisant un modèle de performance, soit a posteriori, par un monitoring du système. Bien souvent des systèmes sont mal réglés, à cause d'un manque d'expertise dans les différents domaines cités

ainsi qu'en raison de la complexité des fonctions d'un administrateur système (responsable du tuning). Ces problèmes suggèrent d'incorporer ces différents domaines d'expertise (analyse des données, détections des problèmes, modélisations,...) dans un système à base de connaissances pour le réglage de performance.

Tuning Aid

Tuning Aid est un système à base de connaissances qui possède suffisamment de connaissances des aspects internes de UNIX, de l'analyse des données et des modèles de performance pour diagnostiquer les problèmes de performance, faire des recommandations de tuning et inférer de nouvelles recommandations à partir d'anciens succès. Tuning Aid pourrait selon ses "développeurs" être intégré à un système expert d'administration générale d'un centre informatique qui contiendrait en outre un expert opérateur, un expert réseau et un expert sécurité... nous rediscuterons cette perspective de façon plus générale au point 4.7.

Tuning Aid constitue selon nous un projet quantitativement bien plus modeste que YES/MVS mais non moins intéressant. Nous avons choisi de décrire ce système car il s'adresse à un autre type de problèmes que YES/MVS, tout aussi important dans la perspective d'une automatisation des tâches d'un centre de calcul. De plus, les gains potentiels en terme d'une productivité accrue et d'un service meilleur apparaissent ici plus clairement. Tuning Aid est encore dans une phase de développement mais des tests ont déjà été effectués avec succès. Vu la complexité du processus de tuning d'un système, nous avons volontairement omis les aspects trop techniques dans la présentation qui suit.

4.5.2 Présentation de TUNING AID

Tuning Aid est un système à base de connaissances qui reçoit des données de mesure du système, un historique du tuning et des informations de divers modules d'analyse de performances, en vue de détecter des problèmes et de faire des

recommandations quantitatives. La figure 4.9 décrit l'environnement de TUNING AID; sar, sadp, acctcom,... sont des outils de UNIX (System V) pour la collecte des mesures du système.

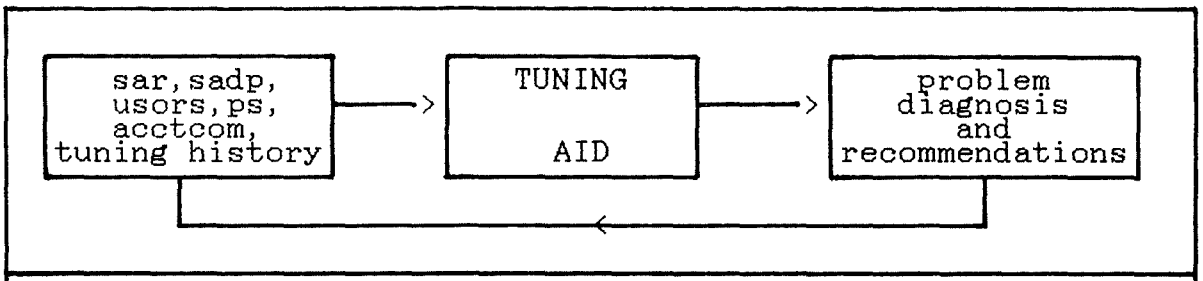

Figure 4.9 : Tuning Aid (SAMADI 1987).

Les règles du système peuvent être classées en trois catégories :

1. Règles générales de tuning décrites dans les manuels de tuning et d'administration du système ou obtenues d'administrateurs système expérimentés. Par exemple,

```
If (the CPU is idle less than 10% of the time)
Then the CPU is highly utilized;
```

2. Règles obtenues en analysant la performance de domaines spécifiques du système d'exploitation comme,

```
If (a drive's utilization is more than 50% and
    any of the file systems on that drive is
    unorganized)
Then recommend performing a dcopy on the unorganized
     file systems;
```

3. Règles obtenues en analysant les données de mesures comme,

```
If (a problem is detected and the CPU is
    highly utilized)
Then make recommendations;

If (a problem is detected and the CPU is not
    highly utilized)
Then issue warnings;
```

Ce qu'il faut considérer comme "highly utilization" ou "unorganized file systems" est déterminé par différentes sources de connaissances. Certaines de ces valeurs sont déterminées par des seuils critiques dans des règles de type 1. D'autres, telle la mesure de désorganisation d'un système de fichiers, sont dérivées de l'analyse de modèles de performances. Un des objectifs futurs de TUNING AID est de permettre au système d'apprendre à partir de l'historique du tuning et de

régler lui-même le système en ajustant ces valeurs.

La structure générale de TUNING AID a été divisée en trois niveaux, la profondeur du niveau indiquant le détail utilisé pour fournir l'information. Pour parcourir ces niveaux, nous allons considérer le problème du nombre de buffers. Pour rappel, l'idée d'un pool de buffers est de réduire le nombre d'accès disque en réutilisant des blocs qui sont disponibles dans le pool et ainsi d'accélérer les opérations I/O pour satisfaire les exigences du CPU. Plus le nombre de buffers est élevé, plus la probabilité de trouver l'information dans la mémoire centrale est élevée. D'un autre côté, plus on ajoute de buffers, plus la mémoire disponible pour les utilisateurs se rétrécit, provoquant ainsi davantage de "paging" ou de "swapping".

Niveau 1

La première étape du développemt de Tuning Aid est l'intégration des règles générales de tuning. A ce niveau, les données produites par "sar" (System Activity Report) sont lues et, suivant les règles de tuning, certains problèmes sont détectés et des recommandations faites. Ces recommandations ne sont cependant pas assez spécifiques et demandent une analyse plus fine :

```
If (the CPU load is high and the wait for IO is high
     and the cache hit ratio is low)
Then recommend increasing the number of buffer;
```

Dans cet exemple, les données montrent que le CPU passe trop de temps à attendre pour des opérations input/output et que cette situation pourrait être améliorée en augmentant le nombre de buffers. Notons que le "hit ratio" est donné par le nombre de blocs trouvés dans le pool des buffers sur le nombre de blocs référencés.

Une première version de ce niveau a d'abord été implémentée en C puis traduite en OPS5. Les raisons de ce second choix sont en tout point semblables à celles invoquées pour YES/MVS.

Niveau_2

Afin d'obtenir des recommandations plus précises, il est nécessaire d'utiliser des méthodes plus complexes, un bon nombre d'informations (notamment quantitatives) ne pouvant être obtenues en suivant simplement des règles générales. A ce niveau, un ensemble de modules fournissent à Tuning Aid une analyse quantitative de divers aspects de UNIX. Chaque module peut être considéré comme une source de connaissances qui utilise une modélisation et une analyse du système, ainsi qu'une connaissance des aspects internes du système d'exploitation. Selon le sujet du module, l'analyse est prédictive ou empirique. En utilisant ces modules, nous pouvons ajouter davantage de règles à la base de connaissances et faire des recommandations quantitatives. Le module "disk buffer sizing" par exemple, analyse la sensibilité de la taille du pool de buffers vis-à-vis de la charge. Nous pourrions dès lors ajouter la règle suivante :

```
If (cache hit ratios are low)
Then analyze effects of increasing number of buffers;
```

Le résultat d'une telle analyse étant un ensemble de paires (x,y) ou "x" est le nombre de buffers supplémentaires et "y" l'amélioration correspondante attendue pour le ratio.

Niveau_3

A ce niveau, la source majeure des informations de Tuning Aid est l'historique. Les données de mesure existent en trop grand nombre et varient trop dans le temps, les modèles ne donnent que des résultats approchés et les seuils critiques sont quant à eux trop rigides pour détecter des problèmes plus subtils. Pour combler ces trous, le système doit pouvoir filtrer et analyser les informations "historiques" concernant les tunings réalisés et les performances obtenues afin de pouvoir en tirer parti dans des situations de charge similaire. Il pourrait ainsi ajuster certains seuils critiques et ajouter dans le processus de décision des paramètres qui n'avaient pas été considérés jusque là. Tuning Aid pourrait par exemple remarquer qu'un phénomène se produit périodiquement pendant un temps très court, qu'il n'y a donc pas lieu de s'alarmer ni

d'acheter du nouveau matériel. Le développement de Tuning Aid n'en est cependant pas encore arrivé à ce stade. Certains modules utilisent bien des informations historiques mais à des fins très limitées.
(SAMADI 1987)

4.6 Représentation déclarative des connaissances contre représentation procédurale

Comme nous avons eu l'occasion de le remarquer jusqu'ici, la représentation des connaissances dans un style déclaratif semble constituer un élément essentiel de toute solution. Les langages de production comme OPS5 intéressent tout particulièrement les "développeurs". Ce succès se justifie apparemment par le désir de trouver une représentation des connaissances facilement réalisable, modifiable et compréhensible à la fois par un humain et par un ordinateur.

La connaissance étant l'élément essentiel de tout système s'adressant à automatiser des tâches qui requièrent une certaine expertise, la question de sa représentation nous a paru particulièrement importante dans le cadre de ce travail. A l'heure où certains parlent d'une "mode" du déclaratif, nous avons voulu essayer de voir en quoi une telle représentation était intéressante dans les domaines qui nous préoccupent, quels en sont les pièges et les limites, quelle en est l'efficacité ? L'expérience gagnée dans le développement de YES/MVS (et, dans une moindre mesure, Tuning Aid) est à cet égard très intéressante.

4.6.1 Intérêts d'une connaissance déclarative

Une connaissance déclarative peut être considérée comme une connaissance dépourvue d'informations de contrôle (particulièrement des informations contrôlant la séquence d'exécution). Un tableur fournit un bon exemple d'un style de représentation déclaratif. Une cellule d'une feuille de calcul peut contenir soit des données, soit une formule exprimée en termes des valeurs d'autres cellules. Il n'y a cependant pas de représentation de l'information de contrôle, c'est-à-dire que

rien n'indique "quand" la valeur de la cellule doit être recalculée. On présume qu'elle est recalculée chaque fois que les paramètres sur lesquels elle porte sont modifiés.

L'intérêt porté à la représentation déclarative provient principalement de deux de ses propriétés (SCHOR 1986)

1. D'une part, une telle représentation est incomplète en ce sens qu'une méthode (ce que nous avons appelé jusqu'ici mécanisme d'inférence) doit être spécifiée en vue de trouver une solution à un problème pour lequel la connaissance est applicable. Dans la mesure où des stratégies générales de résolution peuvent être extraites de la connaissance, l'approche déclarative réalise alors un niveau d'abstraction non négligeable : il devient possible de "cacher" le contrôle procédural et d'obtenir ainsi une représentation facilement modifiable et compréhensible.
2. D'autre part, l'approche déclarative correspond bien au raisonnement humain. Nous avons vu qu'un expert, interrogé sur son expertise, la décrit souvent sous la forme de règles "situation-->action" du genre :

```
If system load is not too heavy
Then allocate more batch initiators
```

Les experts ne disent pas quand il faut appliquer ces règles, elles sont supposées actives en permanence et appliquées quand il le faut. Cette proximité existant entre les règles de production et l'expression des connaissances par un humain semble être la raison majeure du succès remporté par des langages du genre d'OPS5.

Une étude présentée dans (PAU 1986) a piqué notre curiosité. Celle-ci compare quatre schémas de représentation de connaissances selon neuf critères (tableau 4.2). Il n'est pas dans nos intentions de discuter ici le problème général de la représentation des connaissances (problème clé de l'intelligence artificielle s'il en est) mais nous pouvons tout de même constater que les systèmes de production ne constituent pas le nec plus ultra ! Bien sûr, de nombreux critères, comme l'existence sur le marché de systèmes implémentant ces schémas, ont été omis. Néanmoins, l'auteur conclut que des exigences d'effi-

cacité, de puissance d'expression, de concision et de maintenance aisée dictent le choix de schémas de représentation basés sur des "frames" plutôt que sur des règles de production. Nous ne pouvons malgré tout pas nous empêcher de penser, à la lumière des exemples que nous avons donnés, que pareille conclusion ne peut être tirée indépendamment du domaine d'application.

Features	Frames	Semantic Networks	Predicate Calculus	Production System
Expressiveness	10	10	2	2
Structural Representation	8	10	2	2
Computational Efficiency	10	5	3	3
Modifiability	10	1	1	1
Conciseness	10	2	2	2
Representational Uniformity	8	5	10	10
Logical Consistency	5	5	10	5
Easy Retrieval and Access	10	5	5	5
Multiple Level of Representation	10	10	4	4

Tableau 4.2 : Comparaison de schémas de représentation de connaissances (PAU 1986)

Les développeurs de YES/MVS ont également remarqué des effets de bords très intéressants (MILIKEN 1986):

1. L'utilisation de règles encourage la modularisation dans la mesure où chaque règle contient dans son membre de gauche une description complète des prérequis nécessaires à l'invocation du membre de droite. Elle constitue donc à elle seule un morceau de connaissance. Les "développeurs" essayent de minimiser les interdépendances entre les règles en vue d'une répartition des tâches et d'une maintenance plus aisée, aspects non négligeables dans le cadre d'un projet de la taille de YES/MVS.
2. L'extraction des connaissances est un processus qui ne peut s'envisager que par raffinements successifs. Une représentation déclarative permet d'ajouter facilement des nouveaux morceaux de connaissances alors qu'une approche procédurale

nécessite souvent une réorganisation majeure du logiciel.

3. Les règles de production fournissent un langage pour exprimer des situations contextuelles complexes (membre de gauche) avec des tâches courtes qui sont appropriées dans ce contexte (membre de droite). Si l'on avait utilisé un langage procédural pour automatiser les opérations, une grande partie du code du logiciel résultant aurait été affecté à la recherche de l'activité à exécuter... Le moteur d'inférence fournit à cet égard un mécanisme de dispatching basé sur le contexte. C'est dès lors le contexte qui détermine l'activité à exécuter.

4.6.2 Les pièges du déclaratif

Les pièges les plus courants du déclaratif résultent de l'abstraction existant au niveau du contrôle de l'exécution et impliquent (1) l'exécution de règles de déduction sur des données incohérentes, ou (2) des boucles infinies dans l'application des règles.

La première situation se produit quand il existe des incohérences. Nous avons vu en 4.4.6 un exemple de règle de "truth maintenance" en OPS5. Le problème a été résolu pour YES/MVS en accordant systématiquement aux règles chargées de maintenir la cohérence des données une priorité plus grande qu'aux règles susceptibles d'utiliser ces données incorrectes.

Le second problème survient quand une stratégie, pour corriger une incohérence, provoque directement ou indirectement une autre incohérence qui, lors de sa correction, réintroduit la première incohérence. Ce problème est arrivé fréquemment au cours du développement de YES/MVS, en OPS5.
(SCHOR 1986)

4.6.3 L'efficacité du déclaratif

L'inconvénient majeur du style déclaratif est que la détermination dynamique des règles applicables semble requérir la réévaluation continue des membres de gauche, afin de voir si

les conditions représentées par ceux-ci sont couramment satisfaites. Pour un ensemble de règles très important, il faudrait déjà que les avantages du déclaratif soient remarquables pour justifier ce coût ! Deux techniques ont été utilisées dans l'implémentation de OPS5 pour réduire ce coût :

1. Les différents tests apparaissant dans les membres de gauche sont représentés de façon telle que si 100 règles utilisent un même "pattern" comme par exemple,

   job ¬in-queue true ¬priority high

   le test sera réalisé une fois et partagé par toutes les règles utilisant ce "pattern".
2. L'algorithme "Rete Match" retient les résultats partiels de matching et, au cours d'un cycle, réalise seulement le matching des éléments modifiés de la mémoire de travail avec les membres de gauche. Ceci réduit considérablement les coûts de calcul au prix de la mémoire nécessaire à sauvegarder ces résulats partiels.

(SCHOR 1986)

Notons que plusieurs modifications ont été effectuées sur OPS5 au cours du développement de YES/MVS pour des raisons d'efficacité, comme la compilation des membres de droite des règles (cfr. 4.4.4).

4.6.4 Les limitations du style déclaratif

Le style déclaratif n'est pas toujours préférable au style procédural. Une des raisons en est que le mécanisme de contrôle par défaut fourni avec les langages est parfois tout à fait inefficace comparé à une stratégie de contrôle spécifique. Une autre raison tient au fait que la description d'un problème particulier dans un style déclaratif peut correspondre moins bien à l'approche qu'en aurait naturellement une personne.

Un exemple du premier problème est le tri. De façon déclarative, un tri peut être décrit comme une permutation des éléments en entrée de telle sorte que chaque élément soit ordonné par rapport à l'élément suivant. De nombreux algorithmes procéduraux ont cependant été inventés pour implémenter un tri d'une façon bien plus efficace que cette description déclarative couplée à un mécanisme de contrôle caché. Ce premier problème a été une des motivations principales du développement du langage YES/L1, qui est une tentative de conciliation des intérêts du déclaratif et du procédural (CRUISE 1987).

Un exemple du second problème est la recherche de l'élément minimum ou maximum d'une collection "job" relativement à un attribut "¬cpu-used". Nous pouvons spécifier le job ayant nécessité le plus de temps CPU de façon déclarative comme

> Le job ayant une valeur d'attribut cpu-used'
> tel qu'il n'existe pas d'autre job ayant une valeur
> d'attribut cpu-used" plus grande que cpu-used'

ce qui donnerait en OPS5

```
  ( job ¬cpu-used <job-cpu-used> )
- ( job ¬cpu-used  >  <job-cpu-used> )
```

alors que la plupart des gens auraient trouvé la représentation suivante plus claire :

```
( job ¬cpu-used Maximize )
```

(SCHOR 1984)

Ces deux problèmes ont été parmi les motivations principales du développement de YES/L1 qui est une tentative de conciliation des intérêts du déclaratif et du procédural (CRUISE 1987).

4.7 Perspectives : vers une intégration de différents systèmes experts spécialisés

Au point 3.8, nous avons donné un aperçu des outils d'automatisation existants et le lecteur peut remarquer qu'il existe (du moins au niveau expérimental) des systèmes experts dans chacun des domaines envisagés. Actuellement, on ne connait pas d'exemple de systèmes experts travaillant ensemble. Une telle perspective serait pourtant intéressante pour résoudre des problèmes très complexes, pour éviter des redondances au niveau des connaissances et des données, et pour assurer une certaine cohérence de l'exploitation. Cette idée nous a semblé particulièrement intéressante dans la mesure où il n'est pas raisonnable actuellement d'envisager une approche globale des problèmes d'exploitation. Le point 4.4 a montré qu'une approche globale des seuls aspects de contrôle opérationnel constituait déjà toute une aventure...

En utilisant la technique du tableau noir ("blackboard paradigm"), différents systèmes experts pourraient travailler ensemble. L'idée à la base (STROEBEL 1985) est assez simple. Les divers systèmes experts spécialisés dans un domaine particulier partagent une structure de données commune appelée "blackboard" (cfr. figure 4.10). Chaque S.E. a un rôle bien défini et communique avec les autres au moyen du tableau noir. Afin de résoudre des problèmes complexes, les S.E. y placent leurs résultats; quand un S.E. est activé, il examine le contenu courant du tableau, y ajoute des données ou en modifie. L'exécution des règles dans chacun des S.E. est asynchrone et si l'on suppose que tous les S.E. utilisent un mécanisme d'inférence 'data driven', un message d'un S.E. (modification quelconque de l'état du tableau noir) peut déclencher l'exécution de règles d'un autre S.E. Le contrôle général de l'exécution des règles est confié à un système superviseur.

Il ne s'agit bien sûr ici que d'une "idée" mais si la faisabilité d'une telle solution reste encore à démontrer, un avantage non négligeable s'impose d'ores et déjà : la possibilité d'envisager une approche fragmentaire des problèmes d'exploitation.

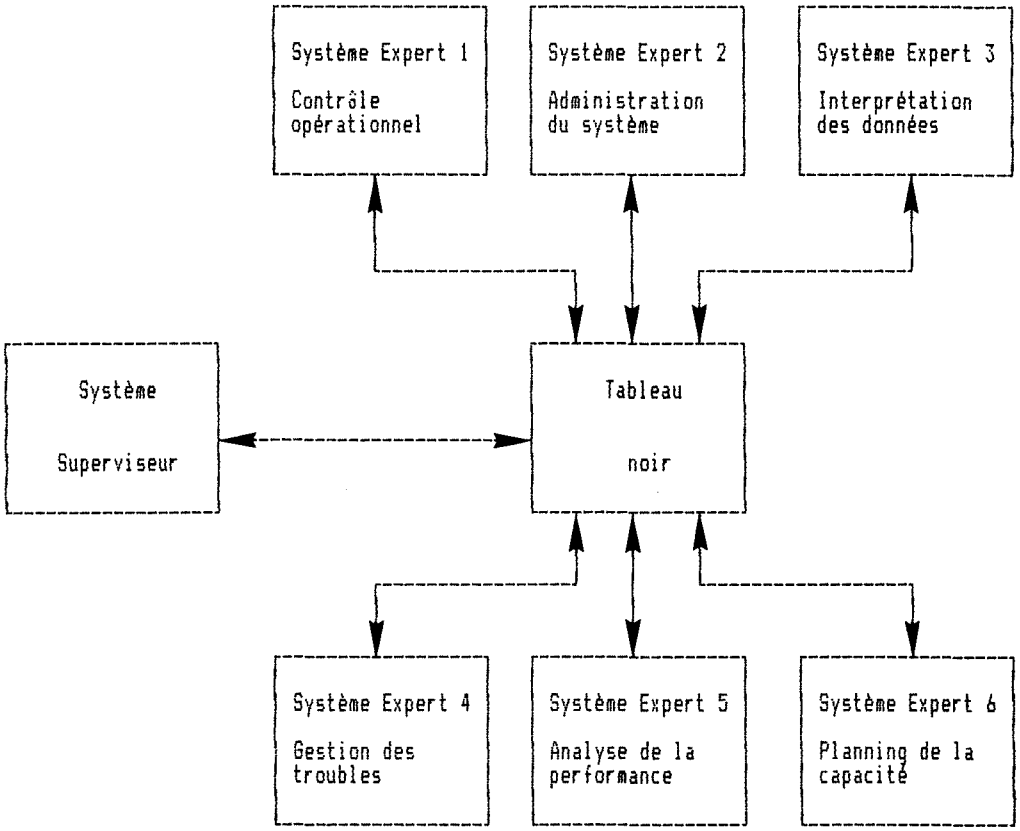

Figure 4.10 : La technique du tableau noir

4.8 Evaluation de l'approche "Système Expert"

Des nombreux reproches adressés aux systèmes experts, beaucoup sont liés à l'état du développement de cette technologie et non à ses principes. De la même façon, nombre de louanges reposent davantage sur les capacités théoriques des S.E. que sur des performances observées. Il faut dès lors être prudent et distinguer l'idéal de l'effectif.

Idéalement, dans la mesure où ils sont bien conçus et implémentés, les systèmes experts garantissent les bénéfices suivants dans les domaines que nous avons considérés.

- La meilleure connaissance possible est toujours disponible dans la base de connaissances. Ce niveau de connaissance est généralement supérieur à celui d'un opérateur, administrateur ou analyste moyen (TERPLAN 1987).

- Les décisions sont effectuées sur base d'une grande quantité de données qui ne seraient pas facilement organisables et compréhensibles par un intervenant humain (TERPLAN 1987).
- Les actions n'exigeant pas d'intervention manuelle sont exécutées à la vitesse de l'ordinateur. Les traitements sont fiables car les S.E. exécutent des tâches répétitives sans interruptions.
- Les S.E. permettent de réduire les besoins en personnel. Celui-ci coopère avec le S.E. et s'occupe seulement des incidents exceptionnels.
- Les réactions aux incidents sont cohérentes et reflètent les politiques en place.
- Les centres sont moins dépendants d'un personnel spécifique car la connaissance et les règles opérationnelles sont codifiées dans la base de connaissances et non dans les "fichiers cachés" du personnel.
- Les S.E. sont flexibles. La base de connaissances est aisément modifiable suite à une évolution du système, des outils ou des conditions opérationnelles.
- Les S.E. peuvent avoir des interfaces avec de nombreux autres outils voire même activer ceux-ci automatiquement (TERPLAN 1987).
- Les S.E. peuvent être utiles à la formation du personnel spécialisé.

Ces différents avantages doivent être relativisés en tenant compte des problèmes inhérents au stade encore quelque peu expérimental de la technologie des S.E.

- La plupart des approches dépendent encore d'un matériel et d'un logiciel particulier, ceci probablement à cause de la performance de certaines architectures (TERPLAN 1987).
- Les "mainframes" manquent encore de compilateurs vraiment puissants pour les langages d'intelligence artificielle (TERPLAN 1987).
- Les problèmes de la maintenance et des tests n'ont pas encore été résolus de façon satisfaisante.
- L'acquisition des connaissances n'est pas encore automatisée.
- Il est assez rare que les S.E. apprennent à partir de raisonnements et de conclusions antérieurs.

- Il est encore très difficile d'estimer la performance de la plupart des S.E.
- Les utilisateurs potentiels au niveau du management ne peuvent pas encore évaluer correctement les coûts et bénéfices.

Actuellement, des systèmes experts s'adressent à tous les domaines que nous avons envisagés mais il n'existe pas encore de liens entre des S.E. de domaines différents.

Des résultats intéressants ont été produits dans le domaine du contrôle opérationnel, mais la plupart des produits restent des prototypes.

Selon Terplan, si les systèmes experts sont appelés à jouer un rôle très important dans l'exploitation des systèmes informatiques, ce sera au terme d'une évolution en quatre temps (TERPLAN 1987). Dans un premier temps des activités de routine seront programmées afin de faciliter l'évaluation de la performance des systèmes informatiques. Dans un second temps, des S.E. de type "conseiller" assisteront le personnel en place dans ses décisions. Dans un troisième temps, des systèmes experts ayant un haut degré d'automatisation et fournissant des conseils corrects et précis, sont attendus. Finalement, des S.E. ayant des possibilités d'apprentissage, résoudront les problèmes les plus complexes, mais voilà qui n'est pas encore pour demain.

CONCLUSION DE LA DEUXIEME PARTIE

Ces dernières années, de profonds changements ont affecté les centres de calcul. Tout d'abord la mise en oeuvre des ressources informatiques s'est considérablement modifiée suite au développement d'une informatique personnelle. Le système central n'impose plus sa loi mais fait l'objet de nouvelles exigences : les utilisateurs attendent désormais de lui un niveau de service semblable à celui offert par un micro-ordinateur, et des structures de coût plus légères. Parallèlement, une croissance de l'informatisation est également observable de par la stratégie des directions d'entreprise à s'appuyer au maximum sur leur outil "système d'information".

Ces phénomènes ont pris une telle ampleur qu'ils sont en train d'inverser les rapports entre utilisateurs d'informatique et informaticiens, obligeant ces derniers à repenser l'organisation des centres informatiques face à un certain nombre de nouvelles exigences : décentralisation des moyens informatiques, intensification de l'exploitation, sécurité et souplesse d'utilisation, qualité du service et rigueur, productivité, etc. Afin de faire face à la situation actuelle et de maîtriser l'évolution future, les gestionnaires considèrent l'automatisation comme un élément essentiel de toute stratégie.

Du contrôle opérationnel à l'administration du système, en passant par la gestion de la performance et le planning de la capacité, tous les domaines de l'exploitation semblent touchés par une complexité croissante et des conditions de travail déraisonnables. Les erreurs sont fréquentes et coûteuses, les spécialistes rares, et leur formation difficile.

Des outils logiciels spécifiques aux divers domaines de l'exploitation existent actuellement sur le marché ou dans les centres de recherche. Les plus simples, et aussi les plus anciens, sont des moniteurs qui collectent des données relatives au système sur lequel ils tournent. D'autres, plus complexes, automatisent en partie le processus de décision : ils fixent des seuils de performance minimum et, une fois ceux-ci atteints, déclenchent des procédures de correction.

Ces outils ne prennent jamais en compte que certains aspects de l'exploitation et sont bien souvent complémentaires... Un problème d'intégration se pose alors afin d'assurer la cohérence de l'ensemble et d'éviter les redondances au niveau de la collecte, de la mémorisation et du traitement des données. Terplan propose une architecture intégrée pour l'évaluation et le contrôle de la performance d'un système mais celle-ci reste bien théorique (TERPLAN 1987).

De façon générale, ces outils sont insuffisants. Ils fournissent trop d'informations à un exploitant qui ne dispose que d'un temps très limité pour prendre ses décisons. Ils peuvent offrir des réponses prédéterminées dans des situations prédéfinies mais sont incapables de fournir une réponse adaptée à des circonstances inhabituelles. Actuellement, ils sont encore peu employés et les centres de calcul restent très attachés aux procédures artisanales. Toutefois cette situation évolue car non seulement ces procédures sont lentes mais elles sont de plus imprécises, ce qui ne correspond pas tellement à l'air du temps...

Les exploitants ont de plus en plus besoin d'outils alliant la flexibilité et le bon sens d'un spécialiste, à la vitesse d'exécution de la machine. Les systèmes experts semblent à cet égard offrir d'intéressantes perspectives : ils peuvent atteindre un niveau de connaissance supérieur à celui d'un exploitant moyen, intégrer dans leur raisonnement bien plus de données que ne le pourrait un être humain, effectuer des recommandations ou prendre des décisions à la vitesse de la machine et justifier leur raisonnement.

Les divers avantages liés aux S.E. doivent être relativisés en tenant compte des problèmes inhérents à la jeunesse de cette technologie. Premièrement, l'acquisition des connaissances n'est pas encore automatisée et il est très rare qu'un S.E. apprenne à partir de raisonnements et de conclusions antérieurs. Deuxièmement, les problèmes de tests et de maintenance n'ont pas encore été résolus de façon satisfaisante. Troisièmement, il est très difficile d'évaluer la performance d'un S.E. hors du cadre de la recherche. Enfin, si des applications existent dans la plupart des domaines de l'exploitation, l'intégration de S.E. spécialisés autour d'une mémoire de travail commune (tableau noir) est encore une idée très théorique, qui devrait faire l'objet de futurs développements.

Si les perspectives offertes par les S.E. semblent réjouissantes, ce n'est qu'au terme d'une évolution que ceux-ci sont appelés à jouer un rôle vraiment important dans l'exploitation des centres informatiques. Des S.E commenceront par automatiser des activités de routine et conseiller les exploitants (des systèmes de ce type sont déjà commercialisés). Par la suite, ils automatiseront une grande partie du travail des opérateurs, analystes ou administrateurs système et les conseilleront de manière précise et efficace dans les situations exceptionnelles (de tels systèmes existent surtout au niveau expérimental). Ce n'est que bien plus tard que l'on aura des S.E ayant des possibilités d'apprentissage, et capables de résoudre les problèmes les plus complexes.

B I B L I O G R A P H I E

ANSI (1970), **Vocabulary for information processing**, American National Standards Institute Inc., New-York (USA), pp. 24, 39-41, 87, 99, 104, 106.

BACHANT, J.; Mc DERMOTT, J. (1984), "R1 Revisited, Four Years in the Trenches", The AI magazine, Fall 1984, pp. 21-32. *

BEAVER, J.E. (1985), "Software for a smooth-running data center", Computer Decisions, , pp. 128-134.

BECKER, J. (1986), "Just where are all the expert systems applications ?", Expert Systems User, january 1986, pp. 16-19.*

BRANCHEAU, J.C.; VOGEL, D.R.; WETHERBE, J.C. (1985), "An investigation of the information center from the user's perspective", Data Base, Fall 1985, pp. 4-14.

BRAUE, J. (1984), "Capacity Management - Part 1: It keeps you running", Computer Decisions, july 1984, pp. 129-137.

BUCHANAN, B.G. (1986), "Expert Systems : working systems and the research literature", Expert Systems, vol. 3, n° 1, pp. 32-51.

CARPER, I.L.; HARVEY, S.; WETHERBE, J.C. (1983), "Computer Capacity Planning : strategy and Methodologies", Data Base, Summer 1983, pp. 1-13.

CHESS, D.M.; WALDBAUM, G. (1981), "The VM/370 Resource Limiter", IBM Systems Journal (USA), vol. 20, n°4, pp. 424-37.

CRUISE, A.; ENNIS,R.; FINKEL,A.; HELLERSTEIN,J.; KLEIN,D.; LOEB,D.; MASSULO,M.; MILLIKEN,K.; VAN WOERKOM,H.; WAITE,N. (1987), "YES/L1 : Integrating Rule-Based, Procedural and Real-time Programming for Industrial Applications", Proceedings of the Third Conference on Artificial Intelligence Applications, Kissimmee, FL, USA, 23-27 feb. 1987, pp. 134-139, 1987, Washington, DC, USA.

DEITEL, H.M. (1984), **An Introduction to Operating Systems**, Addison-Wesley Publishing Company, (USA-CANADA).

DIGITAL (1986 b), **VAX/VMS Operator Student Workbook - part 2**, réf. EY-2281E-LS-0003.

DIGITAL (1986 a), **VAX/VMS Operator Student Workbook - part 3**, réf. EY-2281E-LS-0003.

DUBOIS, P. (1986), "La gestion des centres informatiques, Le pilotage des centres informatiques : des hommes, ou des robots ? Gains de productivité et amélioration des conditions de travail à CISI Télématique", Data Processing : From Discourse to Method. Convention Informatique, 1986, Paris (France), 15-19 sept. 1986, vol. 1, pp. 501-3, 1986, Paris, France.

DUBOIS, Y.; KRAJNC, D. (1988), "Le frein humain", 01 Informatique, n° 1000, pp. XXIII-XXIV.

ELIE, B.; GILLOT, M. (1983), "Contrôle de gestion et critères de qualité d'une direction informatique", Recueil des conférences du printemps. Convention Productivité et Informatique : Pour une entreprise dynamique, Paris (France), 30 mai - 3 juin, pp. 1-3.

ENNIS, R.L.; GRIESMER, J.H.; HONG, S.J.; KARNAUGH, M.; KASTNER, J.K.; KLEIN, D.A.; MILLIKEN, K.R.; SCHOR, M.I; VAN WOERKOM, H.M. (1984), "Automation of MVS Operations, an expert system approach", CMG XV International Conference on the Management and Performance Evaluation of Computer Systems. Conference Proceedings, San Francisco (CA-USA), 4-7 décembre 1984, pp. 234-239, Phoenix (AZ-USA).

ENNIS, R.L.; GRIESMER, J.H.; HONG, S.J.; KARNAUGH, M.; KASTNER, J.K.; KLEIN, D.A.; MILLIKEN, K.R.; SCHOR, M.I; VAN WOERKOM, H.M. (1986 a), "A continuous real-time expert system for computer operations", IBM journal of research and development, vol. 30, n° 1, pp. 14-28.

ENNIS, R.L.; GRIESMER, J.H.; HONG, S.J.; KARNAUGH, M.; KASTNER, J.K.; KLEIN, D.A.; MILLIKEN, K.R.; SCHOR, M.I; VAN WOERKOM, H.M. (1986 b), "Automation of MVS Operations; an expert-systems approach", Computer Systems Science and Engineering, vol.1, n°2, pp. 119-124.

FELDT, T. (1986), "can AI relieve mainframe bottlenecks ?", High Technology, vol. 6, n° 10, pp. 58-60.

GANASCIA, J-G. (1985), "La conception des systèmes experts", La Recherche, n° 170, pp. 142-51.

GEMIS, P. (1985), "Automating Machine Room Operations", Proceedings of Anniversary Meeting 1985. User Friendly Computing, Zurich (Suisse), 22-27 septembre 1985, pp. 135-149.

GRALLA, S. (1987), "An expert System for VM performance analysis", Proceedings of the SEAS Spring Meeting 1987 : Systems Architecture, Montpellier (france), 6-10 avril 1987, pp. 345-351, 1987, Nijmegen (Pays Bas).

GTE's Computer Systems Technology Group (1984), "Capacity Management - Part II: we've never run out of gas", Computer Decisions, july 1984, pp. 138-145.

GUILFOYLE, C. (1986 a), "IBM starts to play the knowledge games", Experts Systems User, October 1986, pp. 18-20.

GUILFOYLE, C. (1986 b), "When the computer asks: 'What's up ?", Expert Systems User, May 1986, pp. 12-15.

GUILFOYLE, C. (1987), "Entering dp through the side door", Expert Systems User, April 1987, pp. 8-10.

GUYNES, J.L. (1988), "Impact of System Response Time on State Anxiety", Communications of the ACM, vol. 31, n° 3, pp. 342-347.

HERIARD DUBREUIL, S. (1986), "L'exploitation doit devenir une véritable profession", Le Monde Informatique, n° 234, pp. 22-23.

HUNT, B. (1986), "Getting the right response", Data Processing, vol. 28, n°6, pp. 299-300.

IFIP (1971), **IFIP Guide to Concepts and terms in Data processing**, North-Holland PUblishing Company, Amsterdam-London, p° 67.

KARNAUGH, M.; ENNIS, R.L.; GRIESMER, J.H.; HONG, S.J; KLEIN, D.A.; MILLIKEN, K.R.; SCHOR, M.I.; VAN WOERKOM, H.M. (1985), "A computer Operator's expert system", New World of the Information Society. Proceedings of the seventh International Conference on Computer Communication, Sydney, Australia, 30 oct.-2 nov. 1984, pp. 810-815, 1985, Amsterdam (Netherlands).

KASTNER, J.K.; ENNIS,R.L.; GRIESMER, J.H.; HONG, S.J; KARNAUGH,M.; KLEIN, D.A.; MILLIKEN, K.R.; SCHOR, M.I.; VAN WOERKOM, H.M. (1986), "Continuous real-time expert system for computer operations", Data Processing, vol. 28, n° 8, pp. 411-425.

LAMPERT, A. (1985), "Expert Systems get down to business", Computer Decisions, january 15, 1985, pp. 138-144. *

LAROUSSE LEXIS (1977), **Larousse de la langue française**, Librairie Larousse, p° 560.

LMI (1986), "L'exploitation doit devenir une véritable profession", Le Monde Informatique, n° 234, pp. 22-23.

Mc DERMOTT, J. (1982), "R1: A Rule-Based Configurer of Computer Systems", Artificial Intelligence, 19 (1982), pp. 39-88. *

MAYNARD, J. (1982), **Dictionnary of Data Processing**, Butterworths, London (UK), p° 62.

MILLIKEN, K.R.; CRUISE, A.V.; ENNIS, R.L.; FINKEL, A.J; HELLERSTEIN, J.L.; LOEB, D.J.; KLEIN, D.A.; MASSULO, M.J.; VAN WOERKOM, H.M.; WAITE, N.B. (1986), "YES/MVS and the automation of operations for large computer complexes", IBM Systems Journal, vol. 25, n° 2, pp. 159-180.

MINGER, R. (1986), "Le pilotage du centre informatique IBM à ORLEANS", Data Processing : From Discourses to Method. Convention Informatique, 1986, Paris (France), 15-19 septembre 1986, vol. 1, p. 504, 1986, Paris (France).

MONNIN, P. (1985), "Combien vaut un informaticien ? Les spécialistes système et méthodes d'exploitation", Le Monde Informatique, n° 200, pp. 20-21.

MOREAU, R. (1988), "L'informatique se personnalise et se latéralise", 01 Informatique, n° 1000, pp. XVII-XVIII.

PAU, L.F. (1986), "Survey of expert systems for fault detection, test generation and maintenance", Expert Systems, vol. 3, n° 2, pp. 100-111.

PILCH, J.; DEESE, D.R.; ARTIS, H.P.; ... (1984), "Defining the performance management function", CMG XV International Conference on the Management and Performance Evaluation of Computer Systems. Conference Proceedings, San Francisco (CA-USA), 4-7 décembre 1984, pp. 425-426, Phoenix (AZ-USA).

POLIT, S. (1985), "R1 and Beyond: AI Technology Transfer at DEC", The AI magazine, Winter 1985, pp. 76-79. *

RAMAEKERS, J. (1984), **Sécurité des systèmes informatiques**, Notes de cours (texte provisoire), FNDP Namur, Belgique.

RAMAEKERS, J. (1985), **Systèmes d'exploitation**, Notes de cours, FNDP Namur, Belgique.

RIVIERE, A. (1983), "The contribution of automated operating processes in improving reliability", Productivité et Informatique : Pour une entreprise dynamique. Recueil des conférences du printemps, Paris, france, 30 mai - 3 juin 1983, vol. 2, pp. 226-229, 1983, Paris (France).

SAMADI, B. (1987), "A Knowledge-based System for Performance Tuning of the UNIX Operating System", USENIX Association Winter Conference Proceedings Washington, DC, USA, 21-23 january 1987, pp. 110-123, 1987, El Cerrito, CA, USA.

SCHOR, M.I. (1984), "Using Declarative Knowledge Representation Techniques : implementing truth maintenance in OPS5", First Conference on Artificial Intelligence Applications, Denver, CO, USA, 5-7 December 1984, pp. 261-266, 1984, Silver Spring, MD, USA.

SCHOR, M.I. (1986), "Declarative Knowledge Programming : Better Than Procedural ?", IEEE Expert, vol. 1, n°1, pp. 36-43.

SIEMENS (1987 a), **BS2000 V8.5B System Operator's Guide**, Manuel, no.U2000-J-Z55-2-7600, chap.5.

STROEBEL,G.J. et al. "A capacity planning Expert System for IBM Systems/38", CMG Proceedings, pp. 204-12, 1985.

TAMINE, J. (1983), "Vers la disparition des départements informatiques", Productivité et Informatique : Pour une entreprise dynamique. Recueil des conférences du printemps, Paris, france, 30 mai - 3 juin 1983, vol. 1, pp. 138-143, 1983, Paris (France).

TERPLAN, K. (1987), "Performance Evaluation and Expert Systems", EDP Performance Review, vol. 15, n° 9, pp. 1-9.

WACHSPRESS, W. (1986), "L'ère des automates", Data Processing : From Discourses to Method. Convention Informatique, 1986, Paris (France), 15-19 septembre 1986, vol. 1, pp. 510-513, 1986, Paris (France).

WALMSLEY, A. (1984), "Prop in practice", Proc., Anniversary Meeting 1984 : Distributed Intelligence, Garmisch-Partenkirchen, RFA, 24-28 septembre 1984, pp. 717-735.

Remarque
Les cinq articles marqués de "*" ne sont pas référencés dans le texte de ce mémoire.